NO. 59 APRIL 1983 U.S. Edition $2.50 International Edition $3.00

MICRO

Advancing Computer Knowledge

123+

p 55: MICRO BBS

p 57: BULLETIN BOARDS

p 63: "THE CUSTOM APPLE" $1500

p 90: ACORN 6502:

p 104: AIM ACCESSORIES

p 116: APPLE DRIVE LIKE MONITOR

BIT PAD

GAME PORT

KEY-LOCK

p 117 EPSON BUFFER

The Micro Communications Revolution

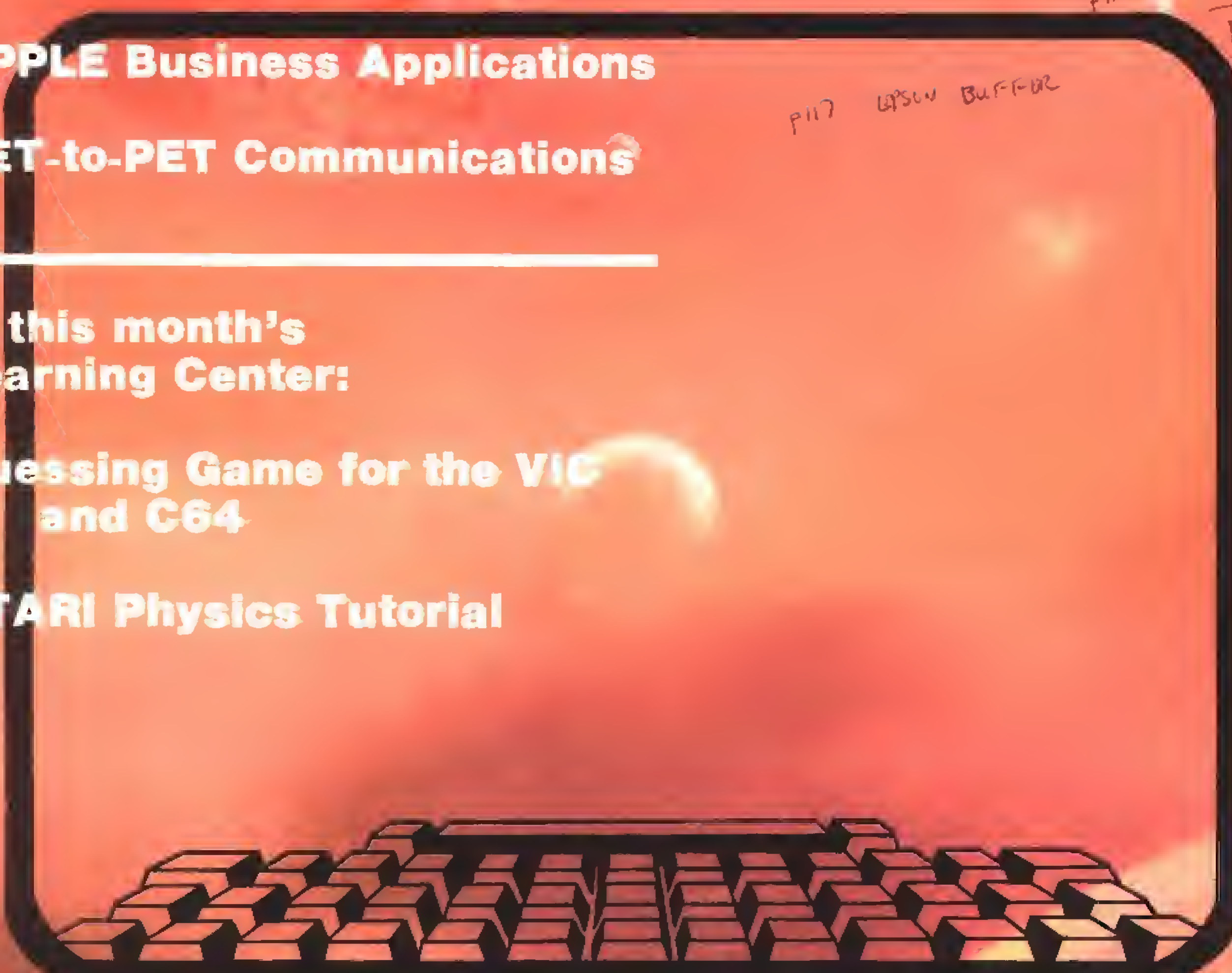
APPLE Business Applications

PET-to-PET Communications

In this month's Learning Center:

Guessing Game for the VIC and C64

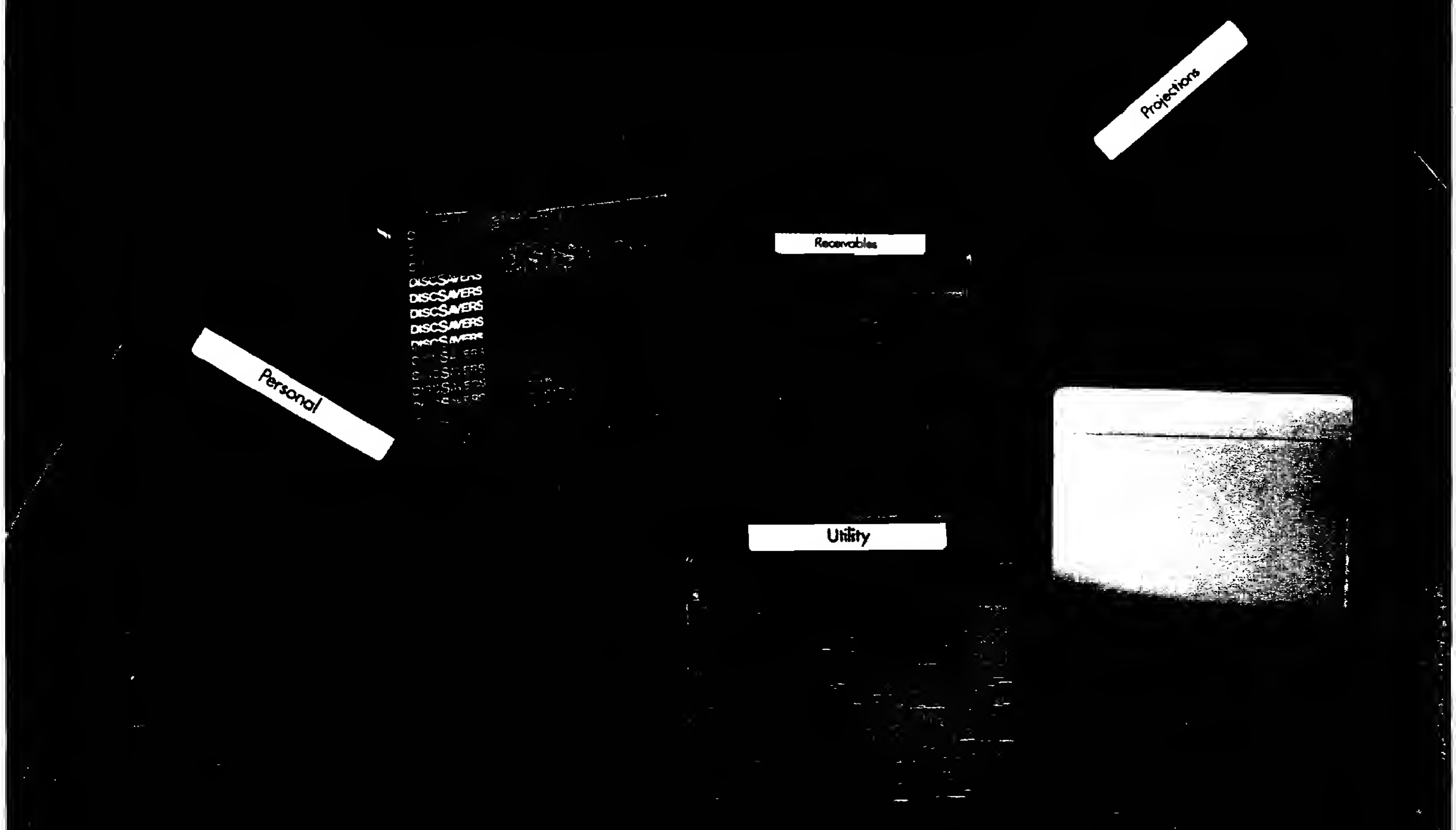
ATARI Physics Tutorial



04
74470 11690

DISCSAVERS

VINYL PROTECTIVE DISK SLEEVES



COLOR CODED: Multi-color DiscSavers™ are designed for easy recognition of individual disks with your own color-keyed filing system. Ideal for office or home use.

PROTECTIVE: Custom grain vinyl provides added protection for magnetic disks by guarding against common handling hazards.

ATTRACTIVE: DiscSavers provide a handsome and professional method of single disk storage and enhance the look of your hardware while protecting your valuable software.

DURABLE: Rigid vinyl construction protects against constant handling to ensure long wear and tear.

PORTABLE: DiscSavers are the only portable vinyl disk sleeves for use with a single diskette that bear the RockRoy mark of quality.

Contact your Dealer or Distributor.

ROCKROY INC.
Computer Products Division

7721 E. Gray Road
Scottsdale, Arizona 85260
(602) 998-1577
Toll-Free 800-528-2361

MICRO™

April Highlights

"**communication** (kə-myōō'nə-kā'shən) *n*... 1. The act of communicating; transmission.... 4. *Plural.* A means of communicating, especially: a. A system for sending and receiving messages, as by mail, telephone, or television."

The American Heritage Dictionary may want to amend this definition to read "...by computer, mail, telephone, or television." Certainly communication by computer offers many possibilities, limited only by our imaginations. "Eight million computer terminals will be in use in American homes by the end of this decade, many linked by information networks to businesses and other data bases," according to J.S. Mayo of Bell Labs (see Bradley Coley's article, "At the Front: The Micro Communications Revolution" p. 26). These terminals will be used for fun and for profit. Subscribers to information banks such as The Source and CompuServe will be able to receive electronic mail, news, weather, and sports; they will be able to teleshop *via* electronic catalogues, and get up-to-the-minute reports on the stock market. Job hunting will become more selective, bartering may return as a form of salesmanship, formal education may revert from the classroom to the home. The possibilities are infinite. Anyone who has a telephone or TV will have access to a world of information through networking.

To learn more about communications and the microprocessor, read Bradley L. Coley's article mentioned above. He presents three theories for what will motivate the interactive and networking potential — home, office, and enterprise. Mr. Coley also discusses the home computer market, networking, and the field of "information for profit." In "Dialing the Networks," (pg. 38) Cliff Glennon maps out the essential steps needed for a MC6809-based home computer to communicate with The Source and CompuServe. He includes a short assembly-language program that implements some basic disk functions, and interfacing and control codes for the MC6850 ACIA. Terry Peterson describes how to turn the Commodore SuperPET into a smart terminal for a mainframe. See "A Not-So-Dumb Terminal Program for the SuperPET" (pg. 31) for a machine-language program that uses the 6551 ACIA serial port for RS-232 I/O.

"PET-to-PET Communications" by F. Arthur Cochrane (pg. 47) provides a machine-language program to transfer an array from one PET to another *via* the user port. And "A Home-Built Communications Interface" by John Steiner (pg. 44) describes how to construct a communications interface. Included is a simple, reliable, and inexpensive design for converting the interface to a telephone modem. "Multi-Microprocessed Tidbits" (pg. 50) shows you how to create a powerful device by running a 6502 and 6809 in the same computer simultaneously. Mike Rosing presents a general description of a specific task for which two processors were used, and discusses some of the problems you might encounter.

The communications section includes an article by our technical editor Phil Daley, who outlines a method MICRO is now using to communicate between the FOCUS, a 6809-based microprocessor (produced by our sister company The Computerist), and the Compugraphic Editwriter 7500. "In-House Communications" (pg. 54) is an informative tutorial that shows you how we use the FOCUS as a text editor, sending material in its final format to the Compugraphic for output.

About the Cover

The original oil painting by Frank Wyman, *Time in Space*, creates an appropriate feeling of expansion and infinity — the feeling generated by today's communications field.

Photo and painting by:
Frank Wyman
Wyman Art Studio
Lowell, MA 01852
(617) 459-7819

Business Applications

"Mutual Fund Charting for APPLE and OSI," by Ralph H. Green (pg. 98) enables you to make, update, and print mutual fund files on both OSI and Apple computers. The programs are written mostly in BASIC (except for a few commands peculiar to OSI) and are easily transportable to other micros. "Analysis of Bond Quotations on the APPLE," by Donald C. Lewis (pg. 92) computes information about the performance of bonds. Data for these computations are available in the financial section of your newspaper. "LETTER-MASK: A Check-Protecting Algorithm" (pg. 102) is an Applesoft BASIC routine by Barton M. Bauers. In addition to number masking, this routine gives your checks additional security by spelling out the amount.

Learning Center

Our new Learning Center opens the classroom door to discussions of momentum, number conversion, and programming concepts about flags and random numbers. "Conservation of Momentum for ATARI and COMMODORE" by Jerry Faughn (pg. 84) helps the beginning computerist examine the conservation law of momentum as applied to collision problems. "Is a Number a Number?" by Phil Daley (pg. 86) shows you how answers are affected by the base of the numbering system you use. "MASTER for VIC-20 and COMMODORE 64" by Loren Wright (pg. 70) is a simple guessing game for one or two players, based on the popular commercial game, and teaches you about flags and random numbers.

And...

Of particular interest this month is the Information Sheet (pg. 57), which includes a list of Bulletin Boards throughout the US and elsewhere. A note of interest here: We received this list from a California data bank *via* a telephone modem connected to the FOCUS.

We hope you find the April issue of MICRO informative. Read, learn, and communicate! MICRO™

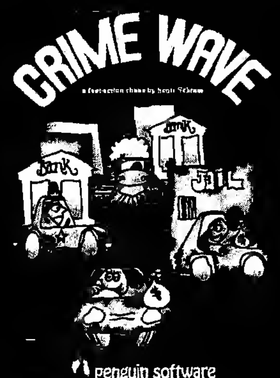
LOOK WHAT YOU CAN GET FOR $19.95!

aos in the bakery
you contend with
antankerous
nveyor belt in an
ort to make pies.

See if you can sneak past the
security guards in this best-seller.

NEW RELEASES

Run a gauntlet of deadly
alien sharpshooters.



The criminal element has run amok
Can you round them up?

We believe games should be fun and that the price of games shouldn't dampen that fun. The growth of the market over the past couple of years leads us to believe that $19.95 may work now as a reasonable game price, so we're trying it for the next six months, and if we're right, longer. This policy does not just apply to new games, but to ALL our games, including our past and current best-sellers! Our bet is that we'll sell more and that the increased sales will offset the decreased income per product. If so, more people get to play our games, and we still make enough to keep developing newer and better software.

As our customers know, at Penguin Software we take a great deal of care and pride in our products. This change in our pricing in no way affects our standards of quality. We pioneered the removal of copy protection from applications software last year in an effort to give you a better product. This year we are trying again to lead the way in putting the customer first.

penguin software *the graphics people*

(312) 232-1984 830 4th Avenue, Geneva, IL 60134

Circle No. 1

Dealer Hotline: (800) 323-0116, retailers only, please.

Available at your local computer store. Dealer and distributor inquiries welcome. Visa MasterCard accepted

MICRO™

Advancing Computer Knowledge

MICRO
34 Chelmsford Street, P.O. Box 6502
Chelmsford, MA 01824 617-256-5515

MICRO is published monthly by: MICRO, Chelmsford, MA 01824. Second Class postage paid at: Chelmsford, MA 01824 and additional mailing offices. USPS Publication Number: 483470. ISSN: 0271-9002. Send subscriptions, change of address, USPS Form 3579, requests for back issues and all other fulfillment questions to MICRO, 34 Chelmsford St., P.O. Box 6502, Chelmsford, MA 01824, or call (617) 256-5515, Telex: 955329 TLX SRVC, 800-227-1617. *Subscription rates (per year):* U.S. $24.00, $42.00 / 2 yr. Foreign surface mail $27.00. Air mail: Europe $42.00; Mexico, Central America, Middle East, North Africa, Central Africa $48.00; South America, South Africa, Far East, Australasia, New Zealand $72.00.

COMMUNICATIONS FEATURE

THE LEARNING CENTER

Entertainment, teleshopping, and home video banking are what the information revolution is all about.
Communications Feature starts on pg. 26.

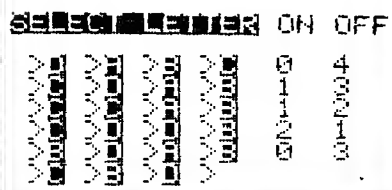
HARDWARE

BUSINESS

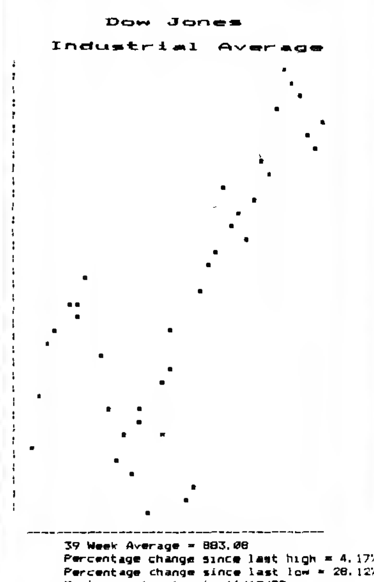
COLUMNS

DEPARTMENTS

Guessing Game pg.70



Fund Charting pg.98



MICRO™

Editorial

MICRO's Learning Center

As you can see by flipping through this month's issue, MICRO is changing — not in content, but in *style*. We're adding more color, more pictures, and more graphics. The only change in content is the addition of The Learning Center, which you'll find beginning on page 67.

Why a new section?

We know the material we offer each month is what you need — serious programming applications and techniques, and pertinent industry news — because you are a serious user. But we also know there are many new users who need tutoring and instruction. We've developed The Learning Center to help these computerists enhance their programming skills.

Many beginners purchase home computers such as the VIC-20, Commodore 64, Atari 400 or 800, TRS-80 Color Computer; most of the articles we publish will run on several of these systems, along with the Apple. We will provide the necessary conversions for running the programs on each machine. For instance, last month "MICROCalc" was offered for all Commodore machines and the Apple.

What will be in The Learning Center?

We plan to offer uncomplicated programs, accompanied by informative text, that will answer your questions about programming. Why were certain lines inserted where they were? What approach is best for writing particular types of programs? What machine offers what characteristics?

Who will read it?

The Learning Center is not an attempt to turn MICRO into a magazine that covers all levels of computing for all levels of users. Instead it allows MICRO to reach the scope of its intended audience: serious, sophisticated users of all levels.

Even advanced users had to start somewhere. Many didn't want to play games or use canned software; they wanted to learn how to develop their own material. We hope readers following The Learning Center will pick up techniques and hints that will advance their programming capabilities and talents.

We'd like to receive feedback from our readers on this new section. Perhaps you have suggestions on topics or approaches. Maybe you could offer ideas on improvements. We would especially like to hear from those who feel they could contribute material to The Learning Center. Write to us soon; help us mold The Learning Center into a valuable and exciting part of MICRO.

Marjorie Morse

Marjorie Morse

MICRO™

MICRO™

Letterbox

OSI Questions

Dear Editor:

I own an OSI C1P series II computer and a Radio Shack Lineprinter VII; this configuration introduces a second linefeed by the printer, therefore double-spacing each printed line.

Apparently Radio Shack computers have an interpreter that doesn't send a linefeed so the printer must provide one. I would appreciate it if your readers could offer some help. This printer performs well and I'd hate to exchange it because of this annoying problem.

Ray Audette
46 Carre Provence
Neufchatel P.Q.
Canada G2B 3R3

Dear Editor:

I've had nothing but trouble with my OSI C4P since the day I bought it. The trouble has been diagnosed as faulty memory. I was wondering if there is a memory program that can check the entire memory and locate the chip that is giving trouble. Also, what do you put in to check existing memory? How can you control the cursor (back up to re-write)?

I've sent two letters to OSI and never received a reply. I can't even trade it for a newer one; even the dealer I bought it from won't take it on trade! Can any of your readers help me with this problem? (And, does anyone want to buy a C4P?).

Jeff Guernsey
112 Overhill
Salina, KS 67401

Readers Help Out

Dear Editor:

A few months ago you published my letter to tell your readers that I was interested in compiling a book of listed programs for use in microcomputer applications in medicine. I received letters from all parts of America, Canada, South America, Europe, Israel, South Africa, and even a letter from China. There were early morning phone calls, picture postcards, packets of discs, bundles of listings; it was a tremem-dous response.

The outcome is that the book is now published by medical Software Co., Box 874, Center Moriches, New York 11934, price $80.00. The volume contains medical application programs for patient scheduling, record retrieval, simple billing, utilization of equipment, simple statistics; standard deviation calculations and curve fitting routines.

Programs are still coming in and are being reviewed for the second volume which should be ready in April 1983. I want to thank everyone again for the tremendous response.

Derek Enlander, M.D.
University Hospital
New York, NY

Updates and Microbes

Spell 'N Fix

There have been some changes in the configuration that affect my review (Spell 'N Fix 55:102). The disk version has been optimized; disk and tape versions are no longer convertible. The new version is slightly faster and is compatible to Color *Scripsit* disk files. Filespecs are now checked before disk access, so you can recover from accidentally mistyping a filename. Lastly, the disk version is available on protected disk, making backups a little more difficult.

John Steiner
Riverside, ND

Data Sheet Bug

Apparently there is a bug in the BASIC decimal to hex number conversion program in the MICRO Data Sheet

(Continued on page 10)

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 60, CHELMSFORD, MA

POSTAGE WILL BE PAID BY ADDRESSEE

MICRO™

34 Chelmsford Street
P.O. Box 6502
Chelmsford, MA 01824

SAVE 20%!!!!

Subscribe to MICRO™

Use This Postage Paid Card to Order the Next 12 Issues of MICRO and SAVE $6.00 Off Newsstand Price!

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 60, CHELMSFORD, MA

POSTAGE WILL BE PAID BY ADDRESSEE

MICRO™

34 Chelmsford Street
P.O. Box 6502
Chelmsford, MA 01824

A Feast Of Computing Ideas...

Order These Books From MICRO™

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 60, CHELMSFORD, MA

POSTAGE WILL BE PAID BY ADDRESSEE

MICRO™

34 Chelmsford Street
P.O. Box 6502
Chelmsford, MA 01824

MICRO™

Reader Service...

For More Information On The Advertisers In This Issue!

INVESTMENT TAX ANALYST
Apple® II Version

Developed by Advanced Investment Strategies, Inc.

Tax planning made simple.

Take the number crunching out of investment strategy, with this easy-to-use tax analyst. Used as a template for VisiCalc®, it lets you instantly project the profitability of any investment (even *series* of investments) for any period up to six years. Calculates "before and after" tax liability and savings... analyzes the impact of tax credits and interest limitations...and calculates net cost, present value, and internal rate of return. For the Apple® II, 64K memory using VisiCalc. One 5¼" disk with documentation. $150.00

USING VISICALC®
Getting Down To Business
Book/Disk Set

Developed by Carol Klitzner & Matthew Plociak, Jr.

You don't have to be a computer wiz to get all the computing power your VisiCalc was designed to deliver.

Now you can exploit *every* time-saving feature and management function built into VisiCalc. This unique book/software package explains all its functions and commands, *and* gives you ready-to-run layouts for financial planning and forecasting, cash flow analysis, inventory management, financial ratios, break-even analysis, and a host of other applications. 288-page book and one 5¼" disk for the Apple II, 48K memory using VisiCalc. $56.90

SCHEDULER/CALENDAR
Apple® II Version

Developed by Ellen Montrose Cohen

Turn your Apple into a tireless, efficient electronic secretary.

A must for every busy professional. Simple, dependable, and easy-to-use, this total time management system lets you enter, find, display, delete, add to, and print an appointment or entire day's schedule in seconds...move appointments when there's a cancellation...interface with other files (such as phone numbers or addresses)... annotate your schedule...even print labels, "reminder" cards, and simple bills. One 5¼" disk with documentation for the Apple II, 48K memory. Available in May. $65.00

MULTIGRAPH
Apple® II Version

Developed by Robert Abey

A powerful, versatile, extremely friendly computer graphics package.

Need to display financial or other numerical data clearly, quickly, and conveniently? Just select the type of graph you want—bar or line, pie charts or scatter diagrams—input the values, and MULTIGRAPH does the rest. Change the values, the graphs change automatically. You can even change from one type of graph to another...and print at the touch of a button. One 5¼" disk with documentation for the Apple II, 48K memory. Available in May. $85.00

GOLDEN DELICIOUS GAMES FOR THE APPLE® COMPUTER
Book/Disk Set

Howard M. Franklin, Joanne Koltnow, LeRoy Finkel

Mind-challenging fun for the Apple® II. Ready-to-run game programs, plus techniques and subroutines more experienced programmers can use to plug into existing games or build new ones. 150-page book plus two 5¼" disks for the Apple II, 32K memory. $47.90

APPLE® BASIC
Data File Programming
Book/Disk Set

LeRoy Finkel and Jerald Brown

How to create and maintain your own data files for billings, inventories, mailing lists, numerical and statistical data, and more—plus ready-to-run file programs. 303-page book plus one 5¼" disk for the Apple II, 48K memory. $34.90

Run with Wiley Professional Software

Unlocking the power of computing.

Ask for Wiley Professional Software at your local computer store. Or order directly from us with this coupon—and use any program free for 15 days.

WILEY PROFESSIONAL SOFTWARE
a division of John Wiley & Sons, Inc.
605 Third Avenue
New York, N.Y. 10158

Apple® is a registered trademark of Apple Computer, Inc. VisiCalc® is a registered trademark of VisiCorp

For faster service **CALL TOLL FREE:** 800-526-5368. In New Jersey, call collect: (201) 797-7809. Order Code 3-9887 VISA, MasterCard, American Express accepted on phone orders.

Mail to: **Wiley Professional Software**
P.O. Box 092
Somerset, N.J. 08873

Please send me the software indicated to use free for 15 days. If not completely satisfied, I may return any software within the trial period and owe nothing. (Restricted to continental U.S. and Canada.)

☐ **Payment enclosed plus sales tax.** Wiley pays normal bookrate postage/handling. We normally ship within 10 days. If shipment cannot be made within 90 days, payment will be refunded.

☐ **Bill me.** ☐ **Bill my company.**

☐ (1-88953-9) AIS: **TAX ANALYST** . $150.00
☐ (1-89004-9) Klitzner **VISICALC** $56.90
☐ (1-87459-0) Cohen **CALENDAR** . . $65.00
☐ (1-87460-4) Abey **MULTIGRAPH**. $85.00
☐ (1-89842-2) Franklin **GAMES** $47.90
☐ (1-89843-0) Finkel **APPLE BASIC** . $34.90

Please Print
NAME ____________
FIRM ____________
ADDRESS ____________
CITY ____________
STATE/ZIP ____________
SIGN HERE ____________

CREDIT CARD ORDERS: To charge your order plus local sales tax and shipping/handling, fill in the information below. If you are not satisfied with the book/software set, return it within 15 days for a full credit to your account.

☐ **VISA** ☐ **MASTERCARD**

CARD NO. ____________

____
Expiration date

____
Signature

(Offer valid through Dec. 31, 1983.)
Prices subject to change without notice.

PROFESSIONAL SOFTWARE

3-9887

Circle No. 5

#5 published in the September 1982 issue of MICRO.

As written, the program, run in either Applesoft BASIC or Commodore 4.0 BASIC, returns an @ instead of the initial nine for decimals in the range 36865 ($9001) to 40959 ($9FFF). In fact, the program returns @@@9 for decimal 39321 ($9999).

The following new lines (in place of the existing lines) will correct the bug:

```
50 IF X > = 10 THEN PRINT
   CHR$(X + 55);

60 IF X< = 10 THEN PRINT
   CHR$(X + 48);
```

See listings 1 and 2 for both the original and corrected programs.

Wilmon B. Chipman
Bridgewater, MA

Listing 1

```
5  REM  PUBLISHED VERSION
10  REM  X < 65536
20  INPUT X
30 X = X / 4096
40  FOR J = 1 TO 4
50  IF X > 9 THEN  PRINT  CHR$ (X + 55);
60  IF X <  = 9 THEN  PRINT  CHR$ (X + 48);
70 X = (X -  INT (X)) * 16
80  NEXT J
```

Listing 2

```
5  REM  CORRECTED VERSION
10  REM  X < 65536
20  INPUT X
30 X = X / 4096
40  FOR J = 1 TO 4
50  IF X >  = 10 THEN  PRINT  CHR$ (X + 55);
60  IF X < 10 THEN  PRINT  CHR$ (X + 48);
70 X = (X -  INT (X)) * 16
80  NEXT J
```

Apple Slices Sliced

Two lines in the December Apple Slices column (page 66) were left out. Insert:

```
179 9568 65 9B 179 ADC LOWTR
180 956A 85 9B 180 STA LOWTR
```

Tim Osborn
Manchester, NH

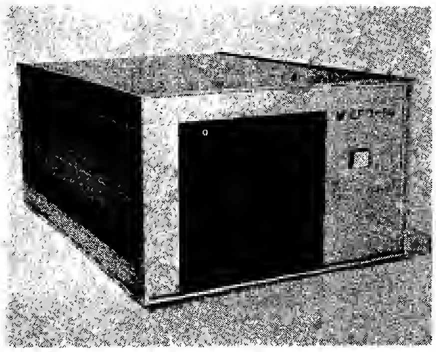
Oops!

In "Print Control for Apple Printers" (58:24), the "#" signs were left off of the following lines of the program.

```
Ø3ØØ A9 Ø4     36  PRNTCTRL LDA #4         * 5 PARAMS (Ø TO 4). COUNT THEM
Ø3ØC C9 2C     41           CMP #COMMA     * NEXT CHR ALSO COMMA?
Ø31D AØ 26     5Ø  CONDOS   LDY #$26       * WHAT IS OUTPUT DEVICE ADDRESS?
Ø328 AØ ØØ     56           LDY #Ø         * THE ADDRESS ITSELF.
Ø335 A9 4D     62           LDA #HOOK      * FINALLY POINT DOS' CSWL ADDRESS
Ø359 C9 8D     79  PRINT1   CMP #CR        * GOT A CARRIAGE RETURN?
Ø379 A9 ØØ     97  PAGETEST LDA #Ø         * COME HERE AFTER CARR.RETURN
Ø388 A9 ØØ    1Ø5  STEPOVER LDA #Ø         * SKIP LINES TO GET TO NEXT PAGE
Ø3A6 A9 C9    12Ø           LDA #TITLE     * GET LBYTE OF TITLE
Ø383 A9 ØØ    125           LDA #Ø         * HIGH BYTE OF PAGE#(SO MAX=255
```

MICRO™

NEW FROM D & N MICRO PRODUCTS, INC.



MICRO-80 COMPUTER

Z80A CPU with 4MHz clock and CP/M 2.2 operating system. 64K of low power static RAM. Calendar real time clock. Centronics type parallel printer interface. Serial interface for terminal communications, dip switch baud rates of 150 to 9600. 4″ cooling fan with air intake on back of computer and discharge through ventilation in the bottom. No holes on computer top or side for entry of foreign object. Two 8″ single or double sided floppy disk drives. IBM single density 3740 format for 243K of storage on each drive. Using double density with 1K sectors 608K of storage is available on a single sided drive or 1.2 meg on a double sided drive. Satin finish extruded aluminum with vinyl woodgrain decorative finish. 8 slot backplane for expansion. 48 pin buss is compatible with most OSI boards. Uses all standard IBM format CP/M software.

Model 80-1200	$2995
2 8″ single sided drives, 1.2 meg of storage	
Model 80-2400	$3495
2 8″ double sided drives, 2.4 meg of storage	
Option 001	$ 95
Serial printer port, dip switch baud rate settings	

Software available in IBM single density 8″ format.

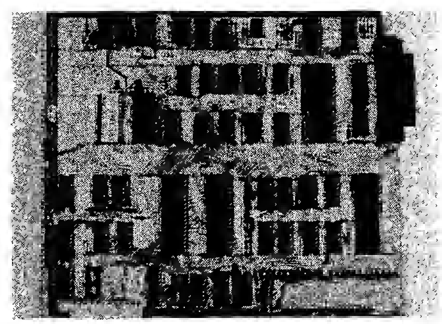
Microsoft	
Basic-80	$289
Basic Compiler	$329
Fortran-80	$410
Cobol-80	$574
Macro-80	$175
Edit-80	$105
Mu Simp/Mu Math	$224
Mu Lisp-80	$174
Digital Research	
PL/1-80	$459
Mac	$ 85
Sid	$ 78
Z-Sid	$ 95
C Basic-2	$110
Tex	$ 90
DeSpool	$ 50
Ashton-Tate	
dBase II	$595
Micropro	
Wordstar	$299
Mail-Merge	$109
Spellstar	$175
Super Sort I	$195
Pascal	
Pascal/MT +	$429
Pascal Z	$349
Pascal M	$355

Convert almost any static memory OSI machine to CP/M® with the D & N-80 CPU Board.

Z80A CPU with 4MHz clock. 2716 EPROM with monitor and bootstrap loader. RS-232 serial interface for terminal communications or use as a serial printer interface in a VIDEO system. Disk controller is an Intel 8272 chip to provide single or double density disk format. 243K single density or 608K double density of disk storage on a single sided 8″ drive. A double sided drive provides 1.2 meg of storage. DMA used with disk controller to unload CPU during block transfers from the disk drives. Optional Centronics type parallel printer port complete with 10 ft. cable. Optional Real Time Calendar Clock may be set or read using 'CALL' function in high level languages. Power requirements are only 5 volts at 1.4 amps. Available with WORDSTAR for serial terminal systems.

INCLUDES CPM 2.2

D & N-80 serial	$695
D & N-80 serial w/Wordstar	$870
D & N-80 video	$695
Option001	$ 80
parallel printer and real time calendar clock	



D & N-80 CPU BOARD

OTHER OSI COMPATIBLE HARDWARE

IO-CA10X Serial Printer Port $125
Compatible with OS-65U and OS-65D software

IO-CA9 Parallel Printer Port $175
Centronics standard parallel printer interface with 10 ft. flat cable

BP-580 8 Slot Backplane $ 47
Assembled 8 slot backplane for OSI 48 pin buss

24MEM-CM9	$380	24MEM-CM9F	$530
16MEM-CM9	$300	16MEM-CM9F	$450
8MEM-CM9	$210	8MEM-CM9F	$380
BMEM-CM9F	$ 50	FL470	$180

24K memory/floppy controller card supports up to 24K of 2114 memory chips and an OSI type floppy disk controller. Available fully assembled and tested with 8, 16, or 24K of memory, with floppy controller (F). Controller supports 2 drives. Needs separated clock and data inputs. Available Bare (BMEM-CM9F) or controller only (FL-470). Ideal way to upgrade cassette based system

C1P-EXP Expansion Interface $ 65
Expansion for C1P 600 or 610 board to the OSI 48 pin buss. Requires one slot in backplane. Use with BP-580 backplane

BIO-1600 Bare IO card $ 50
Supports 8K of memory, 2 16 bit parallel ports may be used as printer interfaces. 5 RS-232 serial ports, with manual and Molex connectors

DSK-SW Disk Switch $ 29
Extends life of drive and media. Shuts off minifloppy spindle motor when system is not accessing the drive. Complete KIT and manual

Disk Drives and Cables

8″ Shugart SA801 single sided $395

8″ Shugart SA851 double sided $585

FLC-8 6 ft. cable from D & N or OSI controller to 8″ disk drive $ 69

5 1/4″ MPI B51 with cable, power supply and cabinet $450

FLC-5 1/4 8 ft. cable for connection to 5 1/4 drive and D & N or OSI controller, with data separator and disk switch $ 75

Okidata Microline Printers

ML82A Dot Matrix Printer $534
120 CPS, 80/120 columns, 9.5″ paper width, friction or pin feed

ML 83A Same as 82A except $895
16″ paper width, 132/232 columns with tractor feed

ML 84 Same as 82A except 200 CPS, $1152
16″ paper width, 132/232 columns, 2K buffer, dot addressable graphics, with tractor feed

D & N Micro Products, Inc.
3684 N. Wells St.
Fort Wayne, Ind. 46808
(219) 485-6414

TERMS $2.50 shipping, Foreign orders add 15%.
Indiana residents add 4% sales tax.

Circle No. 7

Tim Osborn

This month's program, FORMULATE, lets VisiCalc users see the formulas that make up a worksheet all at once rather than one at a time on the edit line. If you do not use VisiCalc, you may still be interested in FORMULATE because it contains a general purpose BASIC subroutine to access individual DISK II sectors (lines 1140-1520).

FORMULATE will take any VisiCalc worksheet file and process it so that all values are stripped out and just the headings and formulas remain. The formulas are translated into headings so they will appear upon loading the file. The data is then saved under the original worksheet's name with ".FORMULAS" appended to the end of the name. The original worksheet file is unchanged, which preserves the data. When the .FORMULAS version of the worksheet is loaded (using the /SL command of VisiCalc), the formulas that make up the worksheet can be viewed all at once along with any headings contained in the worksheet. The ".FORMULAS" version of the worksheet can then be printed using the /P command.

When FORMULATE is run it will display each text file residing on the diskette in the last accessed disk drive one at a time. The user is asked to respond "Y" if the file displayed is the desired file and "N" if it is not. Once the file is selected, FORMULATE will perform its function, notifying the user when the function is completed.

The Program

Lines 5-110 perform an initialization function to get the program ready for operation. Line 120 calls the sector read/write subroutine and reads the VTOC (sector 0, track 17). The subroutine at lines 1210 through 1230 initialize two machine-language subroutines. Line 1430 is a machine-language program to locate the current DOS Input/Output Block (IOB) and place a pointer to the IOB in locations $00 through $01 so that the parameters can be updated by BASIC. Line 1440 is a subroutine that locates the IOB and calls RWTS to perform the operations specified in the IOB.

Line 1235 CALLs the locate-IOB subroutine. Lines 1240-1250 compute the modulo-256 of the buffer address and update the IOB to point to the desired buffer. Lines 1300 through 1390 form a subroutine that takes the desired track (TRK%) and sector (SEC%) and performs the operation specified by OP% (where 1 = Read, 2 = Write). Lines 140-320 read the catalog sectors searching for TEXT files.

Once the user selects a text file to FORMULATE, the program dislays a message "PLEASE WAIT" and begins the main process of the program at line number 450. Line 450 opens the chosen file. Line 460 attempts to delete any .FORMULAS version that may already exist. If the delete function fails because the file does not exist yet then an error-code 6 will be produced ("FILE NOT FOUND"). This condition will be trapped by the ONERR GOTO 880 statement in line 440. Lines 880 through 990 form a general purpose error-handling routine. Error codes 5 and 6 are normal for this program and are handled by the error routine. For error code 6, processing picks back up at line 470. Error code 5 signals an end to the input file so the files are closed and a "FUNCTION COMPLETE" message is displayed.

Line 470 opens the .FORMULAS version. Lines 480 through 870 form the input/output loop where the worksheet is read in, analyzed, and the .FORMULAS file is written out. Lines 500 to 540 replace the normal Applesoft INPUT statement. This is used to avoid the all too familiar "EXTRA IGNORED" problem.

Lines 560 through 750 form a loop, which is used to parse the input record one byte at a time. This loop is an example of finite state automation. It is used here to analyze the worksheet file in order to recognize which records are labels, commands, formulas, and input files that are not worksheet files at all (see line 790).

Lines 760 through 790 check to see in which node (state) the program emerges from the loop. If it emerges in node 6, then the input record was a value (not a computed value or formula). Since FORMULATE strips these from the .FORMULAS version, the program continues to read the next input record without writing anything to the .FORMULAS file.

Line 770 checks for a node 10 or 4, which means that the input record was a label. Since these are written as is, processing continues at the output line number 850. Line 780 checks for a node 8, which means the input record was the VisiCalc Global Column width command (/GC). Since FORMULATE outputs one of these records to the .FORMULAS version at the end of processing (see line 920) to set the columns to the width of the widest formula +1, this record is skipped by jumping to line 480 to get the next input record.

Line 790 checks for a node <> 11, which indicates that the file is not a VisiCalc worksheet; a proper message is displayed and processing is discontinued.

Lines 800 through 840 handle node = 9 (the input record is a formula). These lines simply split the formula into two pieces and place a quote (CHR$ (34)) into the proper position to make the formula a label. Lines 850 through 870 write the record out and jump back to 480 to get the next input record.

(Listing begins on page 14)

```
1 BF = 32744:OP% = 1
5 HM =  PEEK (115) +  PEEK (116) *
     256: REM  SAVE HIMEM
10  HIMEM: BF - 1:HI = 8: REM SET
     HIMEM AND LONGEST FORMULA
20  DIM FL$(105): REM TEXT FILE N
    AME ARRAY
30  GOSUB 1090: REM ESTABLISH ONE
    RR FIX
40  REM *************************
50  REM ** FT%,FL% AND FS% ARE **
60  REM **   OFFSETS INTO THE   **
70  REM **    CATALOG BUFFER    **
80  REM *************************
90 FT% = 13: REM FILE TYPE OFFSET
100 FL% = 14: REM FILE NAME OFFSET
110 FS% = 11: REM FILE STATUS OFFSET
120 TRK% = 17:SEC% = 0: GOSUB 121
     0: GOSUB 1300: REM INIT SECT
     OR R/W ROUTINE AND READ VTOC
130  HOME : HTAB 5: VTAB 5: PRINT
     "PLEASE WAIT - READING CATAL
     OG.......";
140  FOR J = 1 TO 15: REM NUMBER
     OF CATALOG SECTORS
150 TRK% =  PEEK (BF + 1):SEC% =
      PEEK (BF + 2): GOSUB 1300: REM
      READ CATALOC SECTOR INTO BUFFER
160  FOR K = BF TO BF + 210 STEP
     35: REM 7 FILE DESCRIPTORS;
     35 BYTES EACH
170 FS =  PEEK (K + FS%): REM SET
      FILE STATUS CODE
180  IF FS = 0 THEN J = 15:K = BF
      + 210: GOTO 310: REM END LOOP
190  IF FS = 255 GOTO 310: REM SK
     IP DELETED FILE
200 FT =  PEEK (K + FT%)
210  IF  NOT (FT = 0 OR FT = 128)
      GOTO 310: REM SKIP NON-TEXT
      FILES
220 NF = NF + 1: REM  COUNT OF TE
    XT FILES
230 FL$ = "":SP% = 0: REM  INITIA
    LIZE FILE NAME AND TRAILING
    SPACES COUNT
240  FOR L = K + FL% TO K + FL% + 29
250 NW$ =  CHR$ ( PEEK (L))
260  IF NW$ =  CHR$ (160) THEN SP
     % = SP% + 1: GOTO 280
270 SP% = 0: REM RESET TRAILING S
    PACES COUNT
280 FL$ = FL$ + NW$: REM ADD NEW
    CHARACTER TO NAME
290  NEXT : REM  L
300 FL$(NF) =  LEFT$ (FL$,30 - SP
    %): REM  DROP TRAILING SPACE
    S AND SAVE IN FILE NAME ARRAY
310  NEXT : REM  K
320  NEXT : REM  J
330  IF NF = 0 THEN  HOME : PRINT
     "THERE ARE NO TEXT FILES ON
     VOLUME ";: HIMEM: HM: END
340  FOR J = 1 TO NF
350  HOME : HTAB 5: VTAB 5: PRINT
     "IS ";FL$(J);" THE FILE"
360  HTAB 5: VTAB 7: PRINT "YOU D
     ESIRE ?  ENTER Y(ES) OR N(O)
      ";: GET A$
370  IF A$ = "Y" THEN FL$ = FL$(J
     ):J = NF: GOTO 390
380  IF A$ < > "N" GOTO 340
390  NEXT : REM  J
400  IF A$ < > "Y" THEN  HOME : HTAB
     5: VTAB 5: PRINT "NO MORE TE
     XT FILES ON VOLUME": HIMEM:
     HM: END
410 CD$ =  CHR$ (4)
420  HTAB 5: VTAB 9: PRINT "PLEAS
     E WAIT........."
430  POKE 34,10: REM SET TOP OF T
     EXT WINDOW
440  ONERR  GOTO 880
450  PRINT CD$"OPEN ";FL$
460  PRINT CD$"DELETE ";FL$;".FOR
     MULAS"
470  PRINT CD$"OPEN ";FL$;".FORMULAS"
480  PRINT CD$: PRINT CD$"READ ";FL$
490 D$ = ""
500  FOR J = 1 TO 200
510  GET A$
520  IF A$ =  CHR$ (13) THEN LN =
     J - 1:J = 200: GOTO 540
530 D$ = D$ + A$
540  NEXT
550 NODE = 1
560  FOR J = 1 TO LN
570 MD$ =  MID$ (D$,J,1)
580  IF MD$ = ">" AND NODE = 1 THEN
     NODE = 2: GOTO 750
590  IF NODE = 1 AND MD$ = "/" THEN
     NODE = 5: GOTO 750
600  IF NODE = 1 THEN NODE = 11:J
      = LN: GOTO 750
610  IF NODE = 2 AND MD$ = ":" THEN
     K = J:NODE = 3: GOTO 750
620  IF NODE = 2 GOTO 750
630  IF NODE = 3 AND MD$ =  CHR$
     (34) THEN J = LN:NODE = 10: GOTO 750
640  IF NODE = 3 AND MD$ = "/" THEN
     NODE = 4: GOTO 750
650  IF NODE = 3 THEN NODE = 6: GOTO 750
660  IF NODE = 4 AND MD$ = "F" THEN
     NODE = 6:J = J + 1:K = J: GOTO 750
670  IF NODE = 4 THEN J = LN:NODE
      = 10: GOTO 750
680  IF NODE = 5 AND MD$ = "G" THEN
     NODE = 7: GOTO 750
690  IF NODE = 5 THEN J = LN:NODE
      = 10: GOTO 750
700  IF NODE = 6 AND MD$ > "@" AND
     MD$ <  CHR$ (91) THEN J = LN
     :NODE = 9: GOTO 750
710  IF NODE = 6 AND MD$ =  CHR$
     (34) THEN J = LN:NODE = 10: GOTO 750
720  IF NODE = 6 GOTO 750
730  IF NODE = 7 AND MD$ = "C" THEN
     NODE = 8:J = LN: GOTO 750
740  IF NODE = 7 THEN J = LN:NODE = 10
750  NEXT : REM  J
760  IF NODE = 6 THEN  GOTO 480: REM
     SKIP RECORD
770  IF NODE = 10 OR NODE = 4 THEN
      GOTO 850: REM WRITE AS IS
780  IF NODE = 8 THEN  GOTO 480: REM
      SKIP "/GC" - PROGRAM PRODUC
      ES ITS' OWN
790  IF NODE = 11 THEN  POKE 34,0
     : PRINT CD$: HOME : PRINT "T
     HIS DOES NOT APPEAR TO BE A
     WORKSHEET";: PRINT CD$: PRINT
     CD$"CLOSE": PRINT CD$"DELETE
      ";FL$;".FORMULAS": HIMEM: H
     M: END
800  REM  NODE = 9 PASSES HERE
810 L = LN - K
820  IF L > HI THEN HI = L: REM
     SAVE LENGTH OF LONGEST FORMULA
830 LT$ =  LEFT$ (D$,K):RT$ =  RIGHT$
     (D$,L)
840 D$ = LT$ +  CHR$ (34) + RT$
850  PRINT CD$: PRINT CD$;"WRITE"
     ;FL$;".FORMULAS"
860  PRINT D$
870  GOTO 480
880  CALL 768: REM ONERR FIX
890 ER =  PEEK (222): REM SET ERR
     OR CODE
900  IF ER = 6 THEN  PRINT CD$: GOTO
     470: REM NO FORMULAS FILE TO
      DELETE (CONTINUE)
910  POKE 34,0: REM RESET TEXT WI
     NDOW
920  IF ER = 5 THEN  PRINT CD$: PRINT
     CD$"WRITE ";FL$;".FORMULAS":
      PRINT "/GC"; STR$ (HI + 1):
      PRINT CD$: GOTO 1000
930  IF ER = 4 THEN  PRINT "WRITE
      PROTECTED": GOTO 1040
950  IF ER = 10 THEN  PRINT "FILE
      LOCKED": GOTO 1040
960  HOME : PRINT " ERROR CODE =
     ";ER
970  PRINT "IN LINE NUMBER "; PEEK
     (218) +  PEEK (219) * 256
980  IF ER > 15 OR ER = 0 THEN  PRINT
     "SEE PAGE 81 OF THE APPLESOF
     T": PRINT "BASIC PROGRAMMING
      REFERENCE MANUAL": GOTO 1040
990  PRINT "SEE PAGES 114 - 115 O
     F THE DOS MANUAL": GOTO 1040
1000  HOME : HTAB 5: VTAB 5
1010  PRINT "FUNCTION COMPLETED"
1020  HTAB 5: VTAB 7: PRINT "FILE
      ";FL$;".FORMULAS"
1030  HTAB 5: VTAB 9: PRINT "IS N
     OW SAVED ON DISK"
1040  POKE 216,0: REM   TURN OFF
      ONERR GOTO INCASE OF TROUBLE
      W/CLOSE (AVOIDS POSSIBLE LO
      OP)
1050  PRINT CD$: PRINT CD$"CLOSE"
1060  HIMEM: HM: END : REM  RESET
      HIMEM AND END
1070  REM
1080  REM  ***** ONERR FIX *****
1090  FOR J = 768 TO 777: READ K:
      POKE J,K: NEXT : RETURN
1100  DATA  104,168,104,166,223,1
     54,72,152,72,96
1110  REM  *********************
1150  REM ** READ TRACK-SECTOR **
1160  REM **    SUBROUTINE     **
1170  REM ***********************
1180  REM  SEC%=SECTOR TO READ
1190  REM  BF =BUFFER ADDRESS
1200  REM  TRK%=TRACK TO READ
1210  FOR J = 33000 TO 33014
1220  READ I%: POKE J,I%
1230  NEXT
1235  CALL 33000: REM  LOCATE THE IOB
1240 BH% =  INT (BF / 256)
1242 BL% =  INT ((BF / 256 -  INT
      (BF / 256)) * 256 + .05) *  SGN
      (BF / 256)
1244 PTR =  PEEK (0) +  PEEK (1) * 256
1250  POKE PTR + 8,BL%: POKE PTR +
      9,BH%: REM SET BUFFER ADDRESS
1260  RETURN
1270  REM  *********************
1280  REM  * PTR=BEGIN. OF IOB *
1290  REM  *********************
1300  POKE PTR + 4,TRK%
1310  POKE PTR + 5,SEC%
1350  POKE PTR + 12,OP%: REM OPER
      ATION 1 =  READ 2 = WRITE
1360  POKE PTR + 3,0: REM  WILDCARD VOL
1370  CALL 33008: REM CALL LOCIOB+RWTS
1380  POKE 72,0: REM  RESET PREG
1390  RETURN
1400  REM  *******************
1430  DATA 32,227,03,132,00,133,01,96
1440  DATA 32,227,03,32,217,03,96
1450  REM  *******************
1460  REM  * 1ST DATA STMENT *
1470  REM  * MACH. LANG. TO  *
1480  REM  * LOCATE THE IOB  *
1490  REM  * 2ND DATA STMENT *
1500  REM  * LOCATE THE IOB  *
1510  REM  *  AND CALL RWTS  *
1520  REM  *******************
>H10:@SUM(B10...G10)
>G10:+G8-G9
>F10:+F8-F9
>E10:+E8-E9
>D10:+D8-D9
>C10:+C8-C9
>B10:+B8-B9
>A10:"NET
>H9:@SUM(B9...G9)
>A9:"TAXES
>H8:@SUM(B8...G8)
>G8:+G6-G7
>F8:+F6-F7
>E8:+E6-E7
>D8:+D6-D7
>C8:+C6-C7
>B8:+B6-B7
>A8:"GROSS
>H7:@SUM(B7...G7)
>A7:"EXPENSES
>H6:@SUM(B6...G6)
>G6:@SUM(G2...G5)
>F6:@SUM(F2...F5)
>E6:@SUM(E2...E5)
>D6:@SUM(D2...D5)
>C6:@SUM(C2...C5)
>B6:@SUM(B2...B5)
>A6:"TTL SALES
>H5:@SUM(B5...G5)
>A5:"MISC
>H4:@SUM(B4...G4)
>A4:"LABOR
>H3:@SUM(B3...G3)
>A3:"TIRES
>H2:@SUM(B2...G2)
>A2:"BIKES
>H1:"GRAND TTL
>G1:"JUNE
>F1:"MAY
>E1:"APRIL
>D1:"MAR
>C1:"FEB
>B1:"JAN
>H10:"@SUM(B10...G10)
>G10:"+G8-G9
>F10:"+F8-F9
>E10:"+E8-E9
>D10:"+D8-D9
>C10:"+C8-C9
>B10:"+B8-B9
>A10:"NET
>H9:"@SUM(B9...G9)
>A9:"TAXES
>H8:"@SUM(B8...G8)
>G8:"+G6-G7
>F8:"+F6-F7
>E8:"+E6-E7
>D8:"+D6-D7
>C8:"+C6-C7
>B8:"+B6-B7
>A8:"GROSS
>H7:"@SUM(B7...G7)
>A7:"EXPENSES
>H6:"@SUM(B6...G6)
>G6:"@SUM(G2...G5)
>F6:"@SUM(F2...F5)
>E6:"@SUM(E2...E5)
>D6:"@SUM(D2...D5)
>C6:"@SUM(C2...C5)
>B6:"@SUM(B2...B5)
>A6:"TTL SALES
>H5:"@SUM(B5...G5)
>A5:"MISC
>H4:"@SUM(B4...G4)
>A4:"LABOR
>H3:"@SUM(B3...G3)
>A3:"TIRES
>H2:"@SUM(B2...G2)
>A2:"BIKES
>H1:"GRAND TTL
>G1:"JUNE
>F1:"MAY
>E1:"APRIL
>D1:"MAR
>C1:"FEB
>B1:"JAN
/GC16
```

MICRO™

MICRO™

PET Vet

Loren Wright

New Commodore-oriented Books

Without much fanfare the new edition of the *PET/CBM Personal Computer Guide* has arrived from Osborne/McGraw-Hill. The new edition, by Adam Osborne, Jim Strasma, and Ellen Strasma, has been published in two versions, called *PET Personal Computer Guide and CBM Professional Computer Guide*. The emphasis in the PET Guide is on the PET series of computers, the 4022 printer, and the 4040 Guide is on the PET series of computers, the 4022 printer, and the 4040 and 2031 disk drives. The CBM Guide concentrates on the 8032 and the 2001-B, with some mention of the SuperPET and the 8096. Peripherals covered include the 8050 and 8250 disk drives, and the 8024, 8023P, and 8300P printers. Both versions cover the 8010 modem and the 4010 voice synthesizer.

Listings in the PET Guide are presented in upper case/graphics, while the CBM Guide uses mixed case for its listings. More detail is given in the PET Guide on graphics programming, while the CBM pays more attention to numerical calculations and data formatting.

In general, the two books are very similar. They both have the same overall organization, and most of the material is duplicated. Much attention has been paid to updating, correcting, and clarifying material that appeared in the previous edition. One area in particular that received a lot of attention is the section on the CBM relative record system. The second edition of the PET/CBM Guide covered this topic very poorly, including errors and misleading information.

There is also much new material in the new book, including expanded memory maps and detailed information on fixes and upgrades for the various operating systems. In addition to the new material, more program examples are included. Author Jim Strasma offers, at an extra cost, a 'Help' disk, which includes longer demonstration and utility programs. (It also includes "Bennett's Mail List," the subject of Strasma's six-part series in MICRO.)

The two books can serve both as tutorial texts for newcomers and as valuable references for more experienced programmers. I did notice a large number of typographical errors. The Strasmas have published errata lists in *The Midnite Paper*, and the next printing of the guides will correct them. With no comprehensive guide available yet for the Commodore 64, the PET Guide should do very well as a stand-in, since the C64's BASIC is the same as PET BASIC 2. It is too bad that Commodore no longer includes a comprehensive guide with its computers. This is one that every PET or CBM user should have.

Although it is published in the US by COMPUTE!, *Programming the PET/CBM* by Raeto Collin West deserves mention here. It is probably the most comprehensive and detailed description of the PET/CBM operating system available. Particular attention is given to how the system works on a machine-language level. Every BASIC command is explained in detail, with examples. Programs are provided to add extensions, such as TRACE and PRINT USING. There is also an extensive, well explained list of ROM routines. This book is not for the newcomer to programming, but I have found it an essential reference—a good companion to one of the Osborne/McGraw-Hill books.

New Commodore 64 Software

C64 software is beginning to arrive so fast that I can't keep up with it. In my June column, I plan to cover word processors, including Script 64 (Richvale Telecommunications, 10610 Bayview Av., Richmond Hill, Ontario L4C 3N8, Canada), WordPro 3 (Professional Software, 51 Fremont St., Needham, MA 02194), and Paper Clip (Batteries Included, 71 McCaul St., Toronto, Ontario M5T 2X1, Canada).

Also received was a C64 version of KMMM Pascal. Author Willi Kusche

(Continued on page 18)

PET VET *(continued)*

has modified the program so that it dispenses with the BASIC ROMs, thereby making 10K extra available for programs. In addition, errors have been corrected, restrictions removed, and new string handling functions added. The programs operate with the C64's serial bus or with the CIE IEEE-488 interface (but apparently not completely with the C64-Link cartridge). As far as I know, it is the only Pascal available that can generate executable machine code. KMMM Pascal is available from AB Computers (252 Bethlehem Pike, Colmar, PA 18915) for $85.

Support for the Commodore 64

Commodore seems to be doing better at supporting the C64 than it has with previous machines. The Programmer's Reference Guide (described in my December column) arrived at dealers at the end of December. Many of the programs I mentioned then, including the sprite editor, character editor, and simple PET emulator, have been placed in the public domain by Commodore, so you should be able to obtain them from a dealer or users' group.

Commodore's New Machines

As you may remember from a few months back, Commodore announced three new computers. These were the P, B, and BX series. It seems now that the P is the only one of these we're likely to see very soon. It is now called the Commodore 128, and I assume it will have the same 128K, expanded keyboard, and color-and-sound features originally announced. At the Consumer Electronics Show in January, Commodore was saying it would appear in 90 to 120 days.

Commodore showed off some other new products at that show in Las Vegas, but their arrival dates are even less certain. One product was a portable 64K machine, compatible with the Commodore 64. This '64 Series' computer will be available in three configurations: 1) with built-in single disk drive and built-in black-and-white monitor, 2) with single dirve and color monitor, and 3) with dual drive and color monitor.

Commodore will soon be selling its own high-resolution color monitor, designed especially for the Commodore 64 and VIC-20, for $299. Other products shown in prototype versions were a hand-held computer. a piano keyboard for the C64, a voice synthesizer cartridge with interchangeable 'voices' and vocabularies, and a touch-screen panel.

Look for my article in next month's "New Wave of Computers" where I will cover the technical details of the Commodore 64, the Commodore 128, and, I hope, the 64 series portable computers.

TPUG Conference--May 14-15

The Toronto PET User Group (TPUG) is holding a large conference at the Castle Loma campus of George Brown College in Toronto the weekend of May 14-15. I have accepted an invitation to join Jim Butterfield, Steve Punter, Jim Strasma, and a number of other PET experts as a speaker. The presentations will cover a wide variety of topics and experience levels. In addition to the presentations, there will be a major copy session of the TPUG library, which now exceeds 100 disks. Finally, there will be commercial displays, including those from all the stores in the local Toronto area. For more information, write TPUG, c/o Chris Bennett, 381 Lawrence Avenue W., Toronto, Ontario M5M 1B9, Canada.

Lincoln College Summer Computer Seminar

Lincoln College in Lincoln, IL is running a week-long seminar June 19-26. Faculty will include Jim and Ellen Strasma, Jim Butterfield, Len Lindsay, Keith Peterson, and a number of other experts on Commodore equipment. The cost, including room, board, and tuition, is $350. If you don't have a Commodore computer you can bring, a limited number of rentals will be available for an additional fee. You will also be able to purchase a VIC for use in the seminar. For more information, write Jim Strasma at 1280 Richland Avenue, Lincoln, IL 62656.

MICRO™

From Here To Atari

Paul S. Swanson

Clearing Up the Rumors

Many rumors are circulating about new products from Atari. I have seen the 1200 — it is *not* just a rumor. There are also indications that another computer will be announced sometime this summer. But the rumors concerning a 48K Atari 600 no longer look credible since Atari is more likely to bring out a more advanced product; a computer that is between the 400 and 800 is not a step forward. Any statements not officially announced by Atari are probably inaccurate. I'll keep you up to date on significant official announcements.

Missing the Right Cartridge Slot?

I have received a few letters from Atari 400 owners concerned about the right cartridge slot on the Atari 800. Since the Atari's newest computer, the 1200, has no right cartridge slot, and there is very little existing software that requires it, it is not likely that much future software will require it.

Another popular topic in letters is assembly language on the Atari. The Atari uses 6502 assembly language, the same used by Apple, Commodore, and others, so general 6502 books will be useful. A few topics concerning assembly language are specific to the Atari, so if your concerns deal with them, write a letter to me describing the specific application.

For example, one reader asked about creating a cassette bootable program. If you hold down the START button while you power up the computer, it will attempt to load and run a program from tape. The program on tape must be in machine language, which is where assemblers become important. Cassettes are a little more difficult to deal with than disks, primarily because there is no cassette operating system comparable to the disk operating time.

To create a cassette bootable program, you must under stand what the computer does when it reads such a file. The steps that the computer executes in reading the file are:

1. The first record loads into the cassette buffer and the computer stores the first six bytes and saves them in various places. The first byte is not used. The second byte contains the number of records to load. Bytes 3 and 4 contain the address to start saving the program. The last two bytes are the initialization address.
2. The first record (apparently including the first six bytes) is moved from the cassette buffer into the indicated start address, then the rest of the records are read and placed in sequential memory locations following the first record.
3. The computer JSRs to the address of the byte immediately following the first six bytes (starting address plus six). You can use this to load more records into memory if you wish. Return by using an RTS command after clearing the carry (if there was no error), or setting the carry to indicate that there was an error during this routine.
4. The computer next JSRs to your initialization address (indirectly through bytes 5 and 6). In this routine, do whatever initialization you want, then place your actual starting address in DOSVEC (at $000A). Use another RTS to end this routine.
5. Finally, the computer JMPs indirectly through DOSVEC to begin your application. At any time during the execution of your application, SYSTEM RESET is pressed and steps 4 and 5 are repeated.

There is a small bug in Atari's cassette boot routine. At the end of the routine that starts at the start address plus six (step 3), you must stop the cassette motor.

Back to Graphics

Last month I promised some information on using IR modes 4 and 5, which are the character graphics modes that will be available as OS modes 12 and 13 on the Atari 1200. You can, on the 400 and 800, use these two modes if you define your own display list and a custom character set. For hints on how to create a character set, refer to my article in the October 1982 (53:87) issue of MICRO.

There is an important difference in forming each of the characters. You must locate the set on a 1K boundary the same way I describe in the article. However, the formats for each character will be interpreted differently for IR modes 4 and 5. In these modes, the bytes in the set are interpreted as bit pairs, which refer to color registers. Zero refers to the background register, one refers to register zero, two to register one, and three to either register two or register three. In all, you can have up to five colors on the screen with up to four in each character. The reference to registers two or three depends on whether the character is printed in normal or inverse video.

Both of these modes support 40-character lines. Mode 4 uses one scan line per line of format, so it is easily implemented from an IR mode 2 (operating system mode 0) screen, allowing you to access it as if it were a text screen. Mode 5 uses two scan lines per line of format, making it equivalent in resolution to an OS mode 7 map mode screen. You can also modify a text screen for this one, too, but you have only half of the characters available on a full screen, so you must take this into account.

Mixing some IR mode 2 text lines with mode 5 is relatively easy. If you alter the display list to make some of the lines mode 5 and leave others in mode 2, you can

PRINT to the screen as if it were the standard OS mode 0 screen using BASIC. The drawback is that the text lines will use register 2 for the background color and the luminance of register 1 for the letters, so the screen will either have stripes where the text lines go or, if you set register 2 to black, the graphics will have only four colors instead of five, and only three can be used in each graphics character.

Some experimentation with these two modes will explain quite a bit about how they work. I have included a listing at the end of this column that should get you started.

Hardware

If you have a printer that works off the 850 interface, I have one note that may interest you, particularly if you write rather large programs. If the 850 is on when you start up the Atari, some memory is set aside to handle device R:. If you are not using the interface for anything except the printer, you do not need this device, nor do you need to have that extra memory subtracted from your program area. If the 850 is turned on only after the computer is turned on (i.e., the 850 is off when you turn the computer on), this memory is not set aside and device R: will not be available. Device P: is always enabled at power-up, so the printer will be available any time you have both the 850 and the printer turned on.

Reference Books

In a recent column (56:19), I reviewed some reference books that you may want next to your Atari to help with your programming. Since then I learned that Educational Software, Inc. (4565 Cherryvale Avenue, Soquel, CA 95073) publishes references for beginners or experts on the Atari computers, as well as software that will help your programming. Their *Master Memory Map*, for example, is a good roadmap of the hardware and shadow registers in the Atari.

A Closing Note on Character Graphics

When you are finished experimenting with modes 4 and 5, set up a standard text screen and POKE 64, 128, or 192 into location 623. This causes the character set to be interpreted in four-bit groups, effectively implementing a character graphics screen equivalent to OS modes 9, 10, and 11. Note that these are the GTIA modes, so this won't work on the older Atari computers that have CTIA chips instead of the GTIA chips.

Send your letters to Mr. Swanson at 97 Jackson Street, Cambridge, MA 02140.

Here to Atari Listing

```
10 REM ....Character Graphics......
20 REM ....Using IR mode 4.........
30 REM ............................
40 REM ....Paul S. Swanson.........
50 REM ............................
60 REM ****************************
70 REM **Place character set on****
80 REM **a 1K boundary*************
90 REM ****************************
100 DIM X$(1):A=ADR(X$):B=INT(A/1024+1)
    *1024:DIM F$(B-A-1),CSET$(1024)
110 CSET$="@"
120 CSET$(1024)="@"
130 CSET$(2)=CSET$
140 REM ***************************
150 REM **Use CTRL characters for**
160 REM **the redefined characters*
170 REM ***************************
180 RESTORE 1000
190 C=513
200 READ N
210 IF N=256 THEN 300
220 CSET$(C,C)=CHR$(N)
230 C=C+1
240 GOTO 200
250 REM ***************************
260 REM **Declare a GR.0 screen,***
270 REM **then redefine its********
280 REM **display list.************
290 REM ***************************
300 GRAPHICS 0
310 DL=PEEK(560)+PEEK(561)*256
320 POKE DL+3,68
330 I=DL+6
340 N=PEEK(I)
350 IF N=65 THEN 430
360 POKE I,4
370 I=I+1
380 GOTO 340
390 REM ***************************
400 REM **Use standard PRINTs******
410 REM **to display characters.***
420 REM ***************************
430 FOR I=0 TO 26
440 PRINT CHR$(I);
450 NEXT I
460 REM ***************************
470 REM **PRINT the inverse********
480 REM ***************************
490 PRINT :PRINT
500 FOR I=0 TO 26
510 PRINT CHR$(I+128);
520 NEXT I
530 REM ***************************
540 REM **Tell the Atari where to**
550 REM **find the new characters**
560 REM ***************************
570 POKE 756,B/256
580 GOTO 580
960 REM ***************************
970 REM **The custom characters****
980 REM **One DATA per character***
990 REM ***************************
1000 DATA 5,5,5,5,5,5,5,5
1010 DATA 90,90,90,90,90,90,90,90
1020 DATA 175,175,175,175,175,175,175,175
1030 DATA 250,250,250,250,250,250,250,250
1040 DATA 165,165,165,165,165,165,165,165
1050 DATA 80,80,80,80,80,80,80,80
1060 DATA 0,0,0,0,1,5,21,85
1070 DATA 0,0,0,1,5,21,85,86
1080 DATA 0,0,1,5,21,85,86,90
1090 DATA 0,1,5,21,85,86,90,106
1100 DATA 1,5,21,85,86,90,106,170
1110 DATA 5,21,85,86,90,106,170,171
1120 DATA 21,85,86,90,106,170,171,175
1130 DATA 85,86,90,106,170,171,175,191
1140 DATA 86,90,106,170,171,175,191,255
1150 DATA 90,106,170,171,175,191,255,252
1160 DATA 106,170,171,175,191,255,252,240
1170 DATA 170,171,175,191,255,252,240,192
1180 DATA 171,175,191,255,252,240,192,64
1190 DATA 175,191,255,252,240,192,64,0
1200 DATA 191,255,252,240,192,64,0,0
1210 DATA 255,252,240,192,64,0,0,0
1220 DATA 252,240,192,64,0,0,0,0
9999 DATA 256
```

CoCo Bits

John Steiner

Updates

In the December 1982 issue, I presented a short program on a single disk copy routine. A few people have written about a problem with the program crashing in line 200 with a filename error. Other people may run into this problem too, so I will pass along what might be the correction. In program line 130 the routine uses an IF...THEN construct to check for a valid file. If the file does exist and has not been killed, the extension is appended to the filename. A slashbar is also placed in the line as a delimiter; however, an extra space seems to have found its way into the listing. The line should read as follows:

```
130 IF LEFT$ (N$(N),1)  < > CHR$
    (0) AND LEFT$ (N$(N),1)
    CHR$ (255) THEN FI$(K) = N$(N) +
    "/" + EX$(N) : K = K$1
```

The slashbar should be the only character within the quotation marks. The "/" could be replaced with CHR$(47) if you wish. The program would crash in line 200 because the extra space would cause the filename to be one character too long.

I received a letter from Walter Oller of Rapid City, SD, asking about the availability of software capable of handling bowling league team and individual record keeping. If you have software, or are aware of its existence, please let me know.

The "F" Board

Last month I commented on the fact that the TDP System 100 has a slightly different circuit board from the standard CoCo. That statement is no longer true. Since December, Radio Shack has been delivering the computers with this new "F" board. Though the board has no "F" designation on it, it is replacing "E" board computers. If you have a late model Color Computer, you can tell which board you have by lifting the door on the ROM port and looking inside. Computers with an "E" board or earlier have a shield around the processor and memory chips. The shield is almost the only thing visible in the earlier models. "F" board models shield only the RAM chips themselves, so when you look into the port, you can see components all the way through to the other side of the cabinet. The RAM shield is visible to the left of the port as you are looking in.

As I said last month, the computer will probably be offered as a 64K machine. Rumors abound as I am writing this that OS-9 will be available soon in a format licensed to Tandy Radio Shack.

CoCos with 16K are easily converted to 64K. You just have to remove several capacitors, replace the 4116's with 4164's, and move the jumpers from 16K to 64K positions. An additional jumper must be added to the points near the 6821 PIA.

If you have a 32K "F" board, call map type 1, the all RAM mode. The hand-wired modifications required on the earlier boards are no longer necessary.

CoCo Operating Systems

The Radio Shack disk operating system is adequate for BASIC programming and contains many powerful features. There is much to be desired for the machine-language programmer, however. This is partially due to the fact that the system is not well documented. Only a few ROM calls are provided, and sophisticated applications require disassembly of the ROM just to locate and access the routines.

One solution to this can be found in a disk resident DOS. Some commercial programs use the technique, including Radio Shack's own Disk Scripsit. If you can write your own DOS, you will have no problem; but if you are like me, that would be a major hurdle. However, you can purchase disk resident operating systems for the Color Computer. These systems and their utilities give the assembly-language programmer much more power than when using the standard DOS.

I was looking for a disk operating system for quite another reason, however. With many operating systems, files can be read or written by computers using the same DOS, even though they may be different brands. I would be able to send disks along with my articles that contain the text. The editor would then be able to read the file into the text editor for editing and eventual typesetting. The FLEX operating system is one of the more powerful systems available today. In addition, it is implemented on nearly all 6800 series processors. There are several versions of FLEX available, and at least two are implemented on CoCo. I have just purchased Frank Hogg's version and am learning how to use it to full advantage. See November (MICRO 54:23) for a more complete discussion of Frank Hogg's FLEX.

I wish I could say that this month's column was submitted in FLEX format. Unfortunately, a few hours after receiving and loading FLEX, my TDP-100 broke down. But I have already formed some strong impressions on the system in the first few hours.

FLEX is definitely best implemented on at least a two-drive system. I am waiting for a second drive unit, but it has not arrived yet. Working with the system and creating the first backup was enough to convince me that another drive is needed.

One feature of this version of FLEX is DBASIC, a $40.00 program that allows you to use and convert Radio Shack software to FLEX format. The only feature of R/S disk BASIC that is not implemented is random access file capacity. This is not a limitation of FLEX, but of DBASIC. Another appealing feature is a way to call BASIC without accessing Extended BASIC. If you can live without extended BASIC routines, you can use the extra memory

CoCo Bits *(continued)*

(over 39K) for your program.

If you have a monitor, you can use FLEX in a 64 × 32-line format. There are six choices of character and screen dimensions, starting with the standard 32 × 16 format. FLEX is initialized in a 51 × 24 format. A setup program can change that, plus many other power up standards.

There are many people who would like to have the versatilitiy of a DOS but don't have the 64K capacity FLEX requires (or maybe they just don't feel like paying an additional one hundred dollars on a DOS). A viable alternative is Peter Stark's Star-DOS. Star-DOS will run on a 16K CoCo, and requires no modification of the computer. Many of the standad DOS features are implemented, and the user has an opportunity to get the feel of using and programming a disk operating system without spending a lot of money. Star-DOS is priced at $49.95. Unlike FLEX, Star-DOS reads and writes standard Radio Shack format disk files. In addition, a 55-page manual provides all the documentation needed to implement serious disk system applications in assembly language.

Both memory resident and disk resident commands are supported and, like FLEX, it is possible to improve on the DOS by writing your own command routines. If you would like to experiment with a DOS, you might be interested in Star-DOS.

More information is available on these programs from their authors:

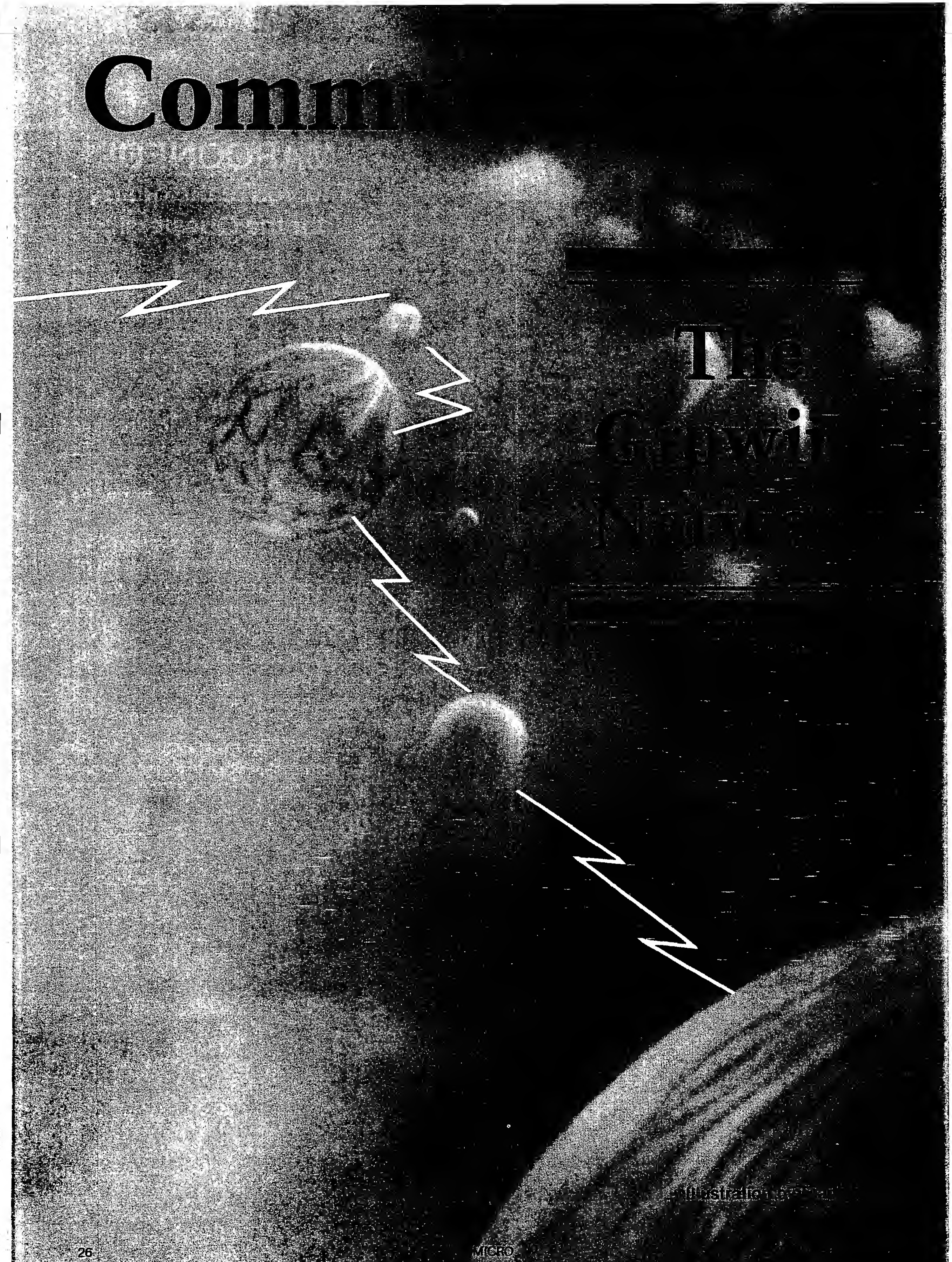
Color Computer FLEX
Frank Hogg Laboratories
The Regency Tower
770 James St.
Syracuse, NY 13203

Star-DOS
Star-Kits
P.O. Box 209
Mt. Kisco, NY 10549

Other disk operating systems are available for the Color Computer from Exatron Corporation and Cer-Comp, among others. I am not familiar with either of these systems. If you have experience with them and would like to pass it along, drop me a line. Next month I will take a closer look at some of the features of a typical DOS.

You may contact Mr. Steiner at 508 Fourth Avenue NW, Riverside, ND 58078.

MICRO


Comm
The

Industry analysts, like their counterparts in economics, have to eat their predictions.

The market for most home computers fell so short of 3-year-old forecasts that, until recently, they called the "home" computer a misnomer, some concluding that there is no market for home computers.

Caught between the "video game gold rush" and the yet-to-come "home banking and teleshopping revolutions," what are we to think about the information society and the micro-on-every-desk predictions? What has become of the visionaries who a few years ago predicted that the eighties would usher in a truly participatory democracy where every home would be plugged into not only Pac Man and the boob tube, but the World Brain, as well?

Electronic lobbying, on-line community organizing, horizontal management, and "People's" data bases were supposed to be up and running by 1983. Technologically, the potential is here, the Utopians insist, but the leadership is not.

Separating fact from wish fulfillment, the predictions may be right about the hardware part of the revolution at least. However "unplugged" we may be as networkers intellectually and politically, the tools of social change are finally proliferating. Estimates for under-$500 home computers show that in 1983 the market is exploding; it should be over $1 billion this year.

"Eight million computer terminals will be in use in American homes by the end of this decade, many linked by information networks to businesses and other data bases," according to J.S. Mayo of Bell Labs. "The nature and location of work and education will thus be dramatically transformed. Eventually, the home/work/study center may replace the classroom and the office for a great many people."

Perhaps, but currently less than 5% of all personal computers sold are connected into any type of network, according to market research surveys. This backs up the industry assumption that current micro users could hardly care less about the personal improvement issues of electronic togetherness; they're into games. What, then, will drive the interactive and networking potential?

Theory One
The Home

The answer can be found by looking at the national investment, all corporate, in what is called "videotex." If you have a TV set or monitor, a telephone, and a connecting micro (better it be a black box with a few buttons saying YES, NO, BUY IT, CHARGE IT), that is called videotex. Major investors in videotex, such as Warner-Amex, Time Inc., CBS, Knight Ridder, have based their development on the assumption that the public will be interested only in "consumer" uses of information. They are convinced that now only entertainment (video games) and, later, teleshopping and home video banking are what the information revolution is all about.

According to one think tank, International Resource Development of Norwalk, Connecticut, one in four U.S. households will have installed in-

However "unplugged" we may be as networkers intellectually and politically, the tools of social change are finally proliferating.

tegrated video terminals and micros capable of accessing outside information by 1990. However, the information will consist simply of news, entertainment and transaction services. Modem-connected terminals, capable as they are of electronic mail and interpersonal networking, will be used primarily for consumption of advertising, news, and teleshopping, according to IRD and their clients, AT&T, GTE, etc., who are "racing to complete trials of new interactive home information."

Just how interactive is this revolutionary technology brought to us by Ma Bell and the corporate providers? It will enable (and I quote IRD) "the customer to use his TV screen as an 'electronic catalogue' on which he can view products and then place his order for them." So much for Ma Bell.

Theory Two
The Office

It's not the home users that will drive the network information marketplace; it's the serious users of information and computer-mediated communications. This school of thought is backed by billions of dollars of vendor advertising and editorials in countless journals. Whether your office is downtown, on campus, or at home, that's the place for plugging into information power.

A recent article in *Personal Computing* entitled "Networking: A Powerful Tool for Personal Communication" catches the eye as you browse the newsstand. Pulse rising, you grab the magazine and read the subtitle: "It may be the most important trend on the horizon of personal computing." On the first page you read: "No longer will an individual computer be limited to its own data resources and computing power; information can be shared quickly, amplified, and amended at will by computer users who might otherwise have to wait for a weekly or a monthly meeting to make the same exchange."

Right on. At last it's being spelled out in print. But wait, the very next sentence says: "A local area network is what makes the power of personal computing for businesses and professionals seem real and practical."

For those who don't know what a local area network is, it means the latest in office automation efficiencies; machines "talking" to other machines, no matter how incompatible. But the incompatibility lies not in the machines themselves; technological advances are taking care of *that* problem rapidly. Senior executives simply see no compelling reason to have micros on their desks. Whether or not they are cyberphobe (afraid of computers) or technophobe (can't wait, can't type), the tried and true ways to communicate are safer and more artistic, suiting the style of upper management — impulsive and unstructured.

For years, office automation professionals have been trying to woo senior management generalists in large organizations, public and private, to their way of thinking. In the seventies, these professionals hyped Management Information Systems, but they flopped — strictly for the technical, DP types. Resurfacing as proponents of Decision Support or IRM (Information Resource Management), they have had no less trouble. Cybernetic missionaries in a pagan land, their ways of improving ex-

ecutive productivity fell on deaf ears.

Yet, the tide seems to be turning. According to Ed Robertson, office automation consultant to the major multinational corporations, "We finally have the technologies... that fit their decision-making style. Number crunching, however graphic and analytical, is not the grabber. Sophisticated ways to communicate with a *wider* Old Boys Network, beyond what they're used to trusting, is what will get them in the water." He adds a caveat: "Only a handful of corporations are managing information at the top less crudely. It will take a few years. In the meanwhile, please don't use the word 'workstation' for CEO offices. At least not to their faces."

Theory 3

Enterprise

If it is not the enlightened home or the liberated office that will be the first to drive the network information marketplace in the next generation (two to three years), then what will?

Although we can see where the obstacles are at the top and the bottom of the power structure, we have only to look at the new wave of micro users to see from where the leadership is coming; the information hungry, the networkers who know they have to unite. Revolutionaries? Utopian Socialists? Hardly.

The Third Wave in networking comes once again from the entrepreneurial "middle" society. The same spirit that pioneered the opening of the West is motivating the opportunity seekers of today. We can see them surfacing in small business, law, accounting, education, medicine, and scientific research. They are people working within corporate structures.

Information for Profit

"On-line entrepreneurs of the world, unite!" may be the rallying cry in a world that is rapidly becoming peopled with opportunity seekers working on their own to market and distribute a wide variety of products and services through self-created networks.

Take the example of two consultants from Arthur D. Little Management Consultant Firm in Boston who were advising clients on electronic publishing and the data base business. These consultants saw a way to make a profit by putting together electronically two groups of people who badly needed each other: hi-tech corporations and techically oriented professionals. Until now, the inefficient job market used classified advertising as its medium for reaching people.

Robert Kvall and George Sacerdote decided to apply their knowledge to this one obvious area of interactive recruiting, using an on-line service over Telenet and Timenet. Last Fall they started Connexions, a Cambridge, Massachusetts, company offering online help-wanted advertising. Job seekers can create a customized resume and send it electronically to the key person in the firm in which they are interested. On the other end, a company can tailor an advertisement by asking certain questions that will further screen the applicants. Only the corporations that the applicant selects get a chance to look at the information.

"Only a handful of corporations are managing information at the top less crudely. It will take a few years."

Most of the major corporations in New England are advertising largely for DP, computer science, or electronic engineers through newspaper ads. Previously, there was no more efficient way. Connexions now makes it possible for both advertisers and subscribers to find each other and pay mutually for the service at each end — with anonymity and confidentiality.

Another successful, on-line, profitable venture is an existing private national association that helps small businesses. The Small Business Science Bureau (SBSB) in Worcester, Massachusetts, has recently established an international computer network in conjunction with the CompuServe Information Service that allows small businesses to send and receive information, electronic mail, software, and data.

A "For Profit" Association

Members benefit from a wide variety of services: volume purchase discounts for products, supplies and health programs, management assistance, and new venture start-up assistance.

Based on a DEC 20 in Worcester, and linked to a gateway to the CompuServe network, a user can send mail to the other 35,000 sucbscribers. SBSB has made available discounted TRS-80s, which include a communications package that acts as both a dumb terminal and also allows one-key transmission of electronic mail and simple transmission of word processing text.

According to Harley Goodwin, VP for Computer Services at SBSB, members will find (and, indeed have already found) ingenious ways to make and save money through the network. Selling the network through cable TV franchises is one; transmitting direct mail lists is another. "We are collectively putting technology to work for small business and the opportunities are endless."

Business Opportunities Network

Another computerized network creating business leads and bringing opportunities together is International Business Opportunities of Woodland Valley, California. IBO collects, screens, and evaluates businesses that are for sale nationally and services would-be investors and buyers. Through their own network of 25 brokers in key cities, potential matchups are referred based on various criteria. For example, if a member broker in New England uses IBO to find a new business in Florida for a buyer, and a member broker in Florida finds a business that fits the bill, both share the commision and pass along a slice to IBO. The company not only maintains the data base by means of continuous search through collective referrals, but it provides full service consulting to both parties, including venture financing.

Many entrepreneurs use The Source Telecomputing Corp. (Source of Silver Springs, MD) and CompuServe Consumer Information Service for communication among close user groups and for fun and profit. These networks continuosly update information of broad public appeal, which can be accessed by any communications micro (dumb or videotex terminal) through local telephone calls. Along with other "information utility" networks, such as Dow Jones or Dialcom, they provide

electronic mail and private storage for a fee. Data bases are accessed by subscribers on a time charge basis. The Source now refers to these closed groups as Private Sectors, and openly solicits sponsors or information providers and groups to set up on-line DBS and electronic mail for publishing activities. The Source will pay royalties for the time your people spend on line. Compuserve calls them SIGs (Special Interests Groups) and publicizes them to attract other entrepreneurial group organizers.

These entrepreneurs are harbingers of things to come. Like the 1890's Gold Rush, the 1980's Information Mine is making money for the lease-holders (providers), the miners (vendors), and those who provide services for the life style that results.

One entrepreneur who does all three is Alan Carr, whose company, Information Inc., is making a profit *via* electronic mail and data base management in a unique way. His company's clients are Fortune 1000 companies and major industry associations that pay him $64 a month per mail box account in return for his building and maintaining an information bank that can be easily accessed through The Source from anywhere. His clients feed him information, internally collected, and he gathers information they specify, externally, whereever he can find it. He's both an information broker and an electronic clipping service.

The end-products include interrogative data bases consisting of personalized material, public opinion, news features, survey highlights, etc. A popular service is the Issues Management file, the latest industry or corporate positions on various issues that management believes affects their organization. In its first year of operation, Information Inc. already has clients spending $5,000-$10,000 a month for the service, depending on the number of subscribers the organization supports.

Information Brokers

Would-be information brokers, on behalf of their clients, can access the Dow Jones News Retrieval Service (a subset of which can be accesssed on The Source and CompuServe). This service has 60,000 subscribers paying $1.20 per prime minute compared to The Source's 30,000 at $.35 and CompuServe's 40,000 at $.38.

Two recent entries into horizontal on-line information services are The Knowledge Index (from Dialogue) and After Dark (from BRS). Between 6:00 p.m. and midnight, for as little as $6 per hour, any personal computer operator with a modem and a password (for a $50 registration fee) can access BRS and get the same in-depth, wide-ranging data files used by BRS Search Service subscribers (Fortune 500 corporations and reference librarians). These include technical and scientific abstracts, medical journals, government studies, business indexes, and general wire service and daily news. A home computer newsletter, electronic mail, shop at home service, and an instant software delivery service all come with the package.

The knowledge Index, from 6:00 p.m. to 5:00 a.m. and weekends, is able to scan more than four million entries from over 10,00 journals and other publications, many updated daily. Computers, electronics, engineering, law, medicine, agriculture, business, psychology, education, and a wide range of information from newspapers, magazines, and government publications are included. You don't get the full articles, only an abstract or summary. The Knowledge Index will take, on line, orders for printed copies of the full text of the articles. Any combination of key words plus any other words, phrases, or numbers that appear in titles, abstracts, author listings, etc., can be used for searching. This raises the search capability of finding specific information beyond that of the conventional information utilities. Connecting words (AND, OR, NOT) enable you to zero in on a topic and find the abstracts of articles dealing with the effects of coffee, sailing in the Straights of Georgia, wind power as an investment, and the effects of stress on managers.

...for as little as $6 per hour, any personal computer operator with a modem and a password can access the same data files used by Fortune 500 corporations.

Videotex and Teletex

In addition to the major networking services, there are videotex and teletex companies offering information over the phone lines and through cable TV. This information is thin news and shopping information and has the advertiser in mind, not the consumer. Teletex offers strictly one-way communication transmitted into the TV set. In some instances you can call up a page and it appears on the TV screen. But you can't go back and find additional information beyond what's in the system. On the other hand, videotex is interactive; you can request information and it is searched and produced.

With ever more valuable, searchable, and specific information services coming on line, the market for them is growing rapidly. Yet it comprises less than a third of personal computer owners and a tiny fraction of the potential population. As this changes, new opportunities are springing up almost daily for those who are discovering that properly mined, refined, and packaged information is money.

Theory E says that enterprise is what will stimulate the network information marketplace in the eighties.

Are you ready?

How do you get an information network started? First find a large, active group that needs to communicate regularly. They may now have a newsletter, publish a calendar or bulletin board, or have an organization that acts as a clearing house for information. Each person should probably have a private network on a dial-up system. Members can have confidential electronic mail and develop data bases, and they can have a window to the outside world and access the popular data bases as well. The network bills the members and will either send you a royalty or you can charge for the content.

Communications Strategies in New York is developing a cooperative start-up venture firm to help launch such enterprises. Dial them at (212) 684-0534. Another source for advice is IncNet. Started by Inc. Magazine for medium-size business owners, the network is currently operating on The Source and Dialcom. So far, it's been an electronic cocktail party because of the lack of leadership. But it could become a hotbed of entreprenerial activity if it gets organized. IncNet operates on a new

computer conferencing software called PARTICIPATE from Participation Systems Inc. (PSI) of Winchester, MA.

Beyond Electronic Mail

PSI's founder, Chandler Harrison Stevens, is associated with the Center for Information Systems Research at MIT. Stevens has long been an advocate of Many-to-Many Communications, his term for the key difference between computer conferencing (CC) and other forms of electronic mail (Telex, facsimile, computer-based messaging, voice store, and forward). What's the difference?

Electronic mail simply provides electronic delivery of fairly ordinary memos that are typed in at one end and come out at the other, or are placed in queue behind other preceding messages. CC allows complex interactions among a group of people by storing the communications on a system, in one place. Any part of the "discussion" can be retrieved at will. You can reconstruct an ongoing "meeting" or correspondence at any time and make comments about specific parts. Many conferences can be held simultaneously, each serving a different purpose, each stored in its own place on the system.

"The single file, lock-step delivery of electronic mail doesn't permit this kind of multi-layered group communication," explains Tom Cross of Cross Communications in Boulder, Colorado. "For the first time, we can begin to really track the progress of a project from inception to completion, allowing software management, new staff, or observers to participate at any point along the way."

Only a small number of corporations, government agencies, and nonprofit organizations are using computer conferencing. For instance, the nation's electric utilities and nuclear equipment suppliers use CC to share experiences and update one another on proposed regulations flowing out of Washington since the Three Mile Island incident.

Ron Simard of the Electric Power Research Institute in Palo Alto, California, has organized CC for the International Nuclear Power Organization (INPO). He claims his is the largest CC in the world: over five hundred people globally. "Subject matter ranges from operating plant experiences and problems, their implications and what to do about them immediately, to government regulation and how to respond," says Simard.

Electronic Jungle Drums

The Bechtel Corportation is using CC to help manage several massive construction projects around the world. One is in the deepest jungles of New Guinea where the largest gold mine in the world was found, along with copper and other valuable minerals. According to Susan Winterstein, coordinator of the project, "The communications between jungle, the managing office in Australia, and our headquarters here in San Francisco would have been a nightmare without computer conferencing. In addition, new people coming on to the project can be quickly updated by retrieving previous entries," she said.

Patricia Pfifer of United Technologies, and ex-telecommunications specialist for AT&T, refers to the research on cost-effectiveness of teleconferencing: "Our studies show that one dollar of teleconferencing equals four dollars of face-to-face meetings and travel." Citing the fact that white-collar workers are the least watched in industry in terms of productivity, the AT&T study concluded that 50% of all business conducted could be through teleconferencing. "It should be, too," Pfifer adds. She cites these advatages:

1. Computer conferencing saves time, not just money (35% reduction in time to achieve the same results).
2. It's convenient. Everyone can follow up on meetings, receive new policies, facts, and product information simulataneously; new people may be added to the conference as needed without briefing; colleagues who would not normally attend the meeting can participate later.
3. It forces discipline (better listening, preparedness, priortizing). The study showed that CC enhances information exchange, briefings, decision-making, problem-solving, and settling differences of opinion. More human than paper, CC makes possible personal support at many levels of the organization.

PROPHET, a large timesharing service, is the central software link that makes possible several joint medical projects now going on at different locations.

Scientists Collaborate

Computers are changing the way scientists communicate. PROPHET, a large timesharing service sponsored by NIH (Biotechnology Resources Program) is the central software link that makes possible several joint medical projects now going on at different locations. Maintained by Bolt, Branek, and Newman (BBN) of Cambridge, Massachusetts, PROPHET allows the researchers to transmit results to investigators elsewhere *via* ARPANET, the research and development network sponsored by the government's Advanced Research Projects Agency. In addition to instant dissemination, it allows the researcher to produce three-dimensional models of molecules and run statistical analyses.

In the Crystal Ball

What's ahead for the micro revolution? To date, what's happening in the home and the office (Pac Man, VisiCalc) is hardly going to change our lives; it's what going to happen that will. Theory 3 (or E for Enterprise) will drive the PC home market as much as all the other incentives (besides entertainment) put together, if the current trend accelerates apace. Electronic cottage industries, as well as electronic publishing by national and regional associations, are springing up so fast that venture capitalists are swamped with investment opportunities.

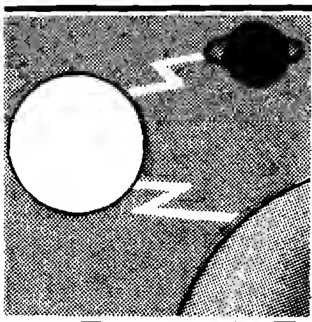
On the office front, local area networks and electronic mail are coming into use and will change the way executives communicate. Whether this will contribute to the Information Society or the Misinformation Society is up to the executives, not the technology.

Bradley "Pete" Coley is founder of Communications Strategies. His firm consults to information technology companies and new ventures. You may contact him at B.L. Coley Public Relations, 533 Second Ave., New York, NY 10016.

MICRO

A Not-So-Dumb Terminal Program for the SuperPET

by Terry M. Peterson



Turn the Commodore SuperPET into a smart terminal for a mainframe with this 6502 machine-language program. The program uses the 6551 ACIA serial port of the SuperPET for RS-232 I/O and requires no external hardware.

Probably the PET's most endearing feature is its convenient screen editor. After I became familiar with this editor, I found ordinary line-oriented text editors all but impossible to use. I felt especially frustrated when using the PET as a dumb terminal to a time-shared computer. Obviously the screen editor is still in there — but how do you use it? Before the advent of the SuperPET I made several attempts to tap this resource in programs designed to work with IEEE RS-232 interfaces, but the results were never satisfying. When I saw the built-in RS-232 port of the SuperPET that uses the 6551 ACIA, I knew the marriage of the PET screen editor and my time-share system was at hand.

This article describes a 6502 RS-232 terminal program that sends edited lines to a host computer using the PET screen editor and the SuperPET's 6551 ACIA. SMARTERM handles conversion of PET-ASCII to true ASCII, as well as control and BREAK characters. The program has an optional character-by-character mode for use with remote screen editors and for other cases when line-by-line mode is undesirable. I've tied the program into the PET's 60 Hz jiffy IRQ interrupt for the input of characters from the host computer, so unexpected input isn't lost. This IRQ patch also allows you to enter from the keyboard, the 8032 special screen formatting characters and send control and BREAK characters. The program does not buffer characters input from the RS-232 port; such buffering is unnecessary for operation up to (at least) 2400 baud as long as the host computer can be made to send several nulls after each carriage return. At 300 baud even the nulls are unnecessary.

The 6551 ACIA makes the programmer's job very easy. This chip takes care of trapping characters at the serial port and decoding them into an 8-bit parallel buffer called the received-data register. The programmer only has to establish such things as the baud rate, parity, and duplexing — and to fetch the bytes from the received-data register before they are overwritten by the next character. Sending characters is even easier — merely POKE the data into the transmit-data register and wait for the 6551 to signal that it has finished sending.

To the 6502 or 6809 in the SuperPET, the 6551 appears as the four memory registers $EFF0 through $EFF3. $EFF0 acts as the receive- or transmit-data register depending on whether it is PEEKed or POKEd. $EFF1 is the status register. It indicates the following: status of the receive and transmit registers; occurrence of parity, framing, and over-run errors; and the status of the RS-232 control lines DCD and DSR. It also contains an IRQ flag (bit 7). If $EFF1 is POKEd with anything the 6551 is reset (i.e., turned off). $EFF2 is called the command register. Most of its bits determine the mode of operation of the 6551 with respect to the microprocessor, but some are used to set the RS-232 parity option. $EFF3, the control register, is used to set the 6551's RS-232 operation with respect to baud rate, word length, and number of stop bits. Table 1 shows the bit settings for the various modes determined by the control and command registers, as well as the bit arrangement of the status register. For further information on the 6551, I recommend the data sheets found in the Synertek 1981-1982 Data Catalog.

Listing 1 shows the assembler source for the terminal program. I have provided extensive comments in the listing, so I will give only a rough outline of the program operation here. (The names in parentheses give the label on the source code line that begins the section described.) The first part of the program (START) is a subroutine that revectors the IRQ through the received-character detect code, sets the necessary 6551 registers, and enables the 6551 IRQ interrupt. If desired, the RTS at the end of this part may be omitted in order to fall directly into the main program loop instead of returning to the calling routine (BASIC).

Next is the main program loop (INLOOP) that handles characters from the keyboard. After the main loop follows (QUIT), the code that restores the IRQ vector and resets the 6551. Next is (CHARIN), the subroutine to

fetch characters from the keyboard. Note that this subroutine alerts the user of char-by-char operation by a non-flashing cursor. (The complication in the code here is setting/clearing control mode for char-by-char output.) Then comes (TSTIRQ), the IRQ vector patch code to trap 6551 IRQ's followed by (INCHR), the code to convert incoming characters to PET-ASCII and, optionally, to display control codes as reversed-field letters. Next is (CTRLTB), a table of the PET-ASCII equivalents for ASCII control codes. Finally there is (KEYTST), the post-jiffy interrupt code that examines each keystroke to test for special screen formatting, control (reverse), and BREAK (STOP) keys. Notice that BREAK is always "live" — that is, even in line-by-line mode the BREAK character is sent while the 'STOP' key is held down.

Listing 2 shows a sample BASIC calling program. Note that this program could be modified to send a log-in sequence between the two SYSs.

If you have machine-language experience and the inclination you could easily extend the terminal program. For example, to add a disk log of your terminal session, take the following steps: 1. Add two JSR $FFD2's to the machine language (one just after the line labeled INLOOP and the other between CHA100 and JMP $E202), 2. OPEN a disk file in the BASIC calling program, and 3. CMD the disk file just before the final SYS into SMARTERM. (This procedure will work even at high baud rates!) To up-load disk files to the mainframe, OPEN the disk file, perform the first SYS, and then GET# bytes from the file, POKE them into 61424, WAIT 61425,16, and loop. Of course this looping could be speeded up if it were implemented in machine language: add an ST check to the main loop and SYS to the sequence LDX #lfn/JSR $FFC6/JMP INLOOP after OPENing the disk file #lfn in BASIC.

Command Register ($EFF2)

Bit(s)	Function
0	Data Terminal Ready (1 = DTR true & rcvr enabled)
1	Receiver IRQ Enable (0 = enabled)
2-3	Transmitter Control
	00 = IRQ disabled, RTS false, Xmitter off
	01 = IRQ enabled, RTS true, Xmitter on
	10 = IRQ disabled, RTS true, Xmitter on
	11 = IRQ disabled, RTS true, Xmit BREAK
4	Echo mode (1 = echo received chars.)
5-7	Parity Control
	XX0 = ignore parity
	001 = odd parity
	011 = even parity
	101 = xmit '1' parity bit, ignore on received data
	111 = xmit '0' parity bit, ignore on received data

Control Register($EFF3)

Bit(s)	Function			
0-3	BAUD rate			
	0000 =	use external generator (not impl. on SuperPET)		
	0001 =	50		
	0010 =	75	1001 =	1800
	0011 =	109.92	1010 =	2400
	0100 =	134.58	1011 =	3600
	0101 =	150	1100 =	4800
	0110 =	300	1101 =	7200
	0111 =	600	1110 =	9600
	1000 =	1200	1111 =	19200
4	Receiver clock (1 = internal gen.)			
5-6	Word length 00,01,10,11 = 8,7,6,5 bits respectively			
7	Stop bits 0,1 = 1,2 stop bits (but see data sheet)			

Status Register ($EFF1)

Bit(s)	Function	
0	Parity error	(1 = error)
1	Framing error	(1 = error)
2	Overrun error	(1 = error)
3	Received data	(1 = true)
4	Transmitted data	(1 = true)
5	(not)DCD (echos pin level, usu. inv. of RS232)	
6	(not)DSR (as DCD)	
7	IRQ (1 = interrupt requested)	

Terry Peterson performs photovoltaic cell research at Chevron Research Company. He first used PETs at work to control and collect data from various laboratory experiments. Now addicted, he writes utility-type software and articles about the PET, CBM, SuperPET, VIC, C64, etc. He may be contacted at 8628 Edgehill Ct., El Cerrito, CA 94530.

FOR YOUR CONVENIENCE THE LISTINGS FOR *SMARTERM* FOLLOW ON THE NEXT THREE RIGHT HAND PAGES

SMARTERM Listing

```
LOC   CODE        LINE
                  ;       S M A R T E R M 5.0  as of 7/25/82
                  ;
                  ;               by T.M. Peterson
                  ;
                  ;  This program allows conversation with a mainframe
                  ;  computer using the built-in screen editor of the
                  ;  SuperPET. The input of characters from the ACIA
                  ;  RS-232 port is interrupt-driven. When location $D3
                  ;  (decimal 211) contains <128, characters are out-
                  ;  put to the ACIA as a line when the operator
                  ;  presses the 'RETURN' key, just as input to a
                  ;  BASIC program is handled. If $D3 contains >127,
                  ;  each character is output when typed.
                  ;  The program also translates 'CBM ASCII' to true
                  ;  ASCII codes and vice versa.
                  ;
                  ;  If desired, incoming control characters may be
                  ;  displayed on the CRT as reverse field letters
                  ;  by storing a 255 in location $D2 (decimal 210).
                  ;  Control characters may be output by typing the
                  ;  'RVS' key, then the letter(s) corresponding to
                  ;  the desired control character(s), and a second
                  ;  'RVS' to exit the 'control' mode.
                  ;  E.g.: [RVS] a [RVS] would send a 'control-A' or
                  ;  $01 character. Pressing the 'RVS' key causes
                  ;  reverse field quote marks to appear on the CRT
                  ;  indicating 'control' mode, but these are not
                  ;  sent over the RS232. (Quotes may be sent by
                  ;  typing TWO quotes, i.e., '""'.)
                  ;
                  ;  Typing 'RVS', 'HOME', 'RETURN' causes a return
                  ;  to BASIC.
                  ;
                        * = 32000        ;Must protect from BASIC

                  SFST   = $98           ;Shift key flag
                  NDX    = $9E           ;Number chars in key buffer
                  CRSPRT = $CD           ;Mode of crsr print
                  CTLPRT = $D2           ;Print cntrl char. flg
                  CHMODE = $D3           ;Char output mode
                  KEYD   = $26F          ;Last kev in buffer
                  ; Point IRQ RAM vector to character input test.
                  ;
7D00  78          START SEI
7D01  A5 90             LDA $90          ;Save IRQ vector
7D03  8D B5 7E          STA JIFFY        ; to restore later
7D06  A5 91             LDA $91
7D08  8D B6 7E          STA JIFFY+1
7D0B  A9 AE             LDA #<TSTIRQ     ;revector IRQ to TSTIRQ
7D0D  85 90             STA $90
7D0F  A9 7D             LDA #>TSTIRQ
7D11  85 91             STA $91
7D13  58                CLI

7D14              ; Initialize 6551 control and command registers.
7D14              ;           /------ 1 stop bit
7D14              ;          / /----- 7 bit word length
7D14              ;         / / /---- internal baud rate gen.
7D14              ;        / / / /--- baud rate = 1200
7D14              ;        ^/\^/^^\
7D14  A9 38             LDA #%00111000  ;set baud, etc.
```

SMARTERM Listing *(continued)*

```
LOC   CODE        LINE
7D16  8D F3 EF          STA $EFF3
7D19              ;             /---- even parity xmit & rcv
7D19              ;            / /--- normal, no echo
7D19              ;           / / /-- xmit IRQ off, xmitr on
7D19              ;          / / / /- enable IRQ on rcv'd data
7D19              ;         /\^ / //- enable rcvr & intrpts
7D19  A9 69             LDA #%01101001  ;turn on 6551 IRQ
7D1B  8D F2 EF          STA $EFF2
7D1E  60                RTS             ;back to BASIC
7D1F              ; Main input loop: Uses PET's screen-edited input
7D1F              ; routine $FFCF to send an edited line all once
7D1F              ; after 'RETURN' if CHMODE<128. Otherwise, $FFE4
7D1F              ; is called to send character-by-character.

7D1F  24 D3       START1 BIT CHMODE      ;char-by-char?
7D21  10 03              BPL INLOOP      ;no
7D23  20 9F 7D           JSR CRSFLP      ;turn on cursor
7D26  20 6A 7D    INLOOP JSR CHARIN      ;get a character
7D29  C9 22              CMP #'"         ;is it quote?
7D2B  D0 07              BNE INL100      ;no, print
7D2D  20 6A 7D           JSR CHARIN      ;get another
7D30  C9 13              CMP #$13        ;quit?
7D32  F0 24              BEQ QUIT
7D34  C9 41       INL100 CMP #$41        ;#/symbol?
7D36  90 08              BCC INL200      ;yes
7D38  C9 5B              CMP #$5B        ;l.c. letter?
7D3A  B0 02              BCS INL150      ;no, u.c.
7D3C  69 20              ADC #$20        ;make ASCII l.c.
7D3E  29 7F       INL150 AND #$7F        ;strip shift bit
7D40  8D F0 EF    INL200 STA $EFF0       ;to 6551

7D43              ; The following delay loop inserts
7D43              ; some dead time between characters
7D43              ; on output. It appears necessary,
7D43              ; at least on the multiplexed timeshare
7D43              ; system I use.

7D43  A9 50              LDA #$50        ;(ca. 25mS)
7D45  8D 45 E8           STA $E845       ;set & start T2
7D48  2C 4D E8    INL260 BIT $E84D       ;wait for T2 timeout
7D4B  50 FB              BVC INL260

7D4D  AD F1 EF    INL300 LDA $EFF1       ;check for send
7D50  29 30              AND #$30        ;done?
7D52  F0 F9              BEQ INL300      ;no
7D54  29 20              AND #$20        ;still have carrier?
7D56  F0 CE              BEQ INLOOP      ; yes


7D58              ; Fall through to here to restore IRQ and quit
7D58              ;
7D58  85 96       QUIT   STA $96         ;set ST (32=lost carrier)
7D5A  78                 SEI             ;restore normal IRQ
7D5B  AD B5 7E           LDA JIFFY
7D5E  85 90              STA $90
7D60  AD B6 7E           LDA JIFFY+1
7D63  85 91              STA $91
7D65  8D F1 EF           STA $EFF1       ;stop 6551 IRQ's
7D68  58                 CLI
7D69  60                 RTS             ;back to BASIC
7D6A              ; Character fetch routine
```

Continued

SMARTERM Listing *(continued)*

```
LDC   CODE       LINE
7D6A             ;
7D6A             ; If CHMODE<128 input is line-by-line,
7D6A             ; otherwise characters are sent as soon
7D6A             ; as they are typed.
7D6A  24 D3      CHARIN BIT CHMODE      ;line-by-line?
7D6C  30 0A             BMI CHGET       ;no
7D6E  20 CF FF          JSR $FFCF       ;wait for RETURN, get char
7D71  C9 0D             CMP #$D         ;EOL?
7D73  D0 38             BNE CHDONE
7D75  4C 02 E2   CHA100 JMP $E202       ;advance crsr

7D78  20 E4 FF   CHGET  JSR $FFE4       ;get char
7D7B  F0 FB             BEQ CHGET       ;got none
7D7D  C9 12             CMP #$12        ;cntrl?
7D7F  D0 06             BNE CHG100
7D81  20 95 7D   CHG050 JSR CRTPRT
7D84  4C 78 7D          JMP CHGET
7D87  C9 92      CHG100 CMP #$92
7D89  F0 F6             BEQ CHG050
7D8B  C9 22             CMP #$22        ;quote?
7D8D  F0 06             BEQ CRTPRT      ;always print
7D8F  A6 CD             LDX CRSPRT
7D91  F0 02             BEQ CRTPRT
7D93  29 1F             AND #$1F
7D95  24 D3      CRTPRT BIT CHMODE
7D97  10 DC             BPL CHA100
7D99  20 9F 7D          JSR CRSFLP      ;flip field
7D9C  20 75 7D          JSR CHA100      ;print char

7D9F             ; Turn on/off cursor at current position

7D9F  48         CRSFLP PHA             ;save char.
7DA0  98                TYA             ; & YR
7DA1  48                PHA
7DA2  A4 C6             LDY $C6         ;flip field
7DA4  B1 C4             LDA ($C4),Y     ; of current
7DA6  49 80             EOR #$80        ; char to show
7DA8  91 C4             STA ($C4),Y     ; cursor position
7DAA  68                PLA             ;restore AC & YR
7DAB  A8                TAY
7DAC  68                PLA
7DAD  60         CHDONE RTS
7DAE             ; Code to handle interrupt-driven input of characters
7DAE             ; from the 6551 ACIA.

7DAE  AD F1 EF   TSTIRQ LDA $EFF1       ;6551 IRQ?
7DB1  10 07             BPL TST60       ;no
7DB3  AE F0 EF          LDX $EFF0       ;get char/clr IRQ
7DB6  29 08             AND #8          ;char waiting?
7DB8  D0 16             BNE INCHR       ;yes
7DBA  2C 13 E8   TST60  BIT $E813       ;jiffy IRQ?
7DBD  10 5D             BPL IRQDDN      ;no, ignore
7DBF  2C 12 E8          BIT $E812       ;clr IRQB
7DC2  58                CLI             ;enable IRQ in case of RCV
7DC3  A9 7E             LDA #>KEYTST    ;point to key test
7DC5  48                PHA             ; before
7DC6  A9 3F             LDA #<KEYTST    ; doing
7DC8  48                PHA             ; 60 HZ IRQ
7DC9  08                PHP             ; as a
7DCA  48                PHA             ; subroutine
7DCB  48                PHA
7DCC  48                PHA
```

SMARTERM Listing *(continued)*

```
LDC   CODE       LINE
7DCD  4C 55 E4          JMP $E455
7DD0             ; Process 6551 IRQ
7DD0             ;
7DD0  8A         INCHR  TXA
7DD1  C9 0D             CMP #$D         ;Treat CR special
7DD3  D0 07             BNE INCH02
7DD5  A9 91             LDA #$91        ;crsr up
7DD7  20 95 7D          JSR CRTPRT      ;(undo auto LF)
7DDA  D0 18             BNE INCH05      ;now do CR

7DDC  C9 20      INCH02 CMP #$20        ;control code?
7DDE  B0 1B             BCS INCH10      ;no
7DE0  24 D2             BIT CTLPRT      ;display cntrl?
7DE2  10 10             BPL INCH05      ;no, print it
7DE4  69 40             ADC #$40        ;make letter
7DE6  48                PHA
7DE7  A9 12             LDA #$12        ;(RVS)
7DE9  20 95 7D          JSR CRTPRT
7DEC  68                PLA
7DED  20 95 7D          JSR CRTPRT
7DF0  A9 92             LDA #$92        ;(OFF)
7DF2  D0 1D             BNE INCH30

7DF4  BD 1F 7E   INCH05 LDA CTRLTB,X    ;get PET equiv.
7DF7  F0 23             BEQ IRQDDN      ;none, never mind
7DF9  D0 16             BNE INCH30      ;do it

7DFB  C9 41      INCH10 CMP #$41        ;#/symbol?
7DFD  90 12             BCC INCH30      ;yes, leave
7DFF  C9 5B             CMP #$5B        ;upper case?
7E01  90 0C             BCC INCH20      ;yes
7E03  C9 7F             CMP #$7F        ;'del'?
7E05  F0 15             BEQ IRQDDN      ;ignore
7E07  C9 60             CMP #$60        ;symbol?
7E09  90 06             BCC INCH30      ;yes
7E0B  E9 20             SBC #$20        ;must be lower case
7E0D  09 80             ORA #$80
7E0F  49 80      INCH20 EOR #$80
7E11  20 95 7D   INCH30 JSR CRTPRT      ;print translat. on CRT

7E14             ; Now fix screen editor pointers to ignore
7E14             ; input from RS232 port.

7E14  A5 C6             LDA $C6         ;bump begin. ptr.
7E16  85 A4             STA $A4
7E18  A5 D8             LDA $D8         ;adj. row ptr.
7E1A  85 A3             STA $A3

7E1C  4C 00 E6   IRQDDN JMP $E600       ;restore regs., do RTI
7E1F             ; Table of PET-control code correspondences
7E1F             ; Zero entry = ignore
7E1F             ;
7E1F  00         CTRLTB .BYTE 0,0,0,0
7E20  00
7E21  00
7E22  00
7E23  00                .BYTE 0,0,0,7
7E24  00
7E25  00
7E26  07
7E27  9B                .BYTE $9B,9,17,$91
7E28  09
7E29  11
```

Continued

SMARTERM Listing *(continued)*

```
LOC   CODE        LINE
7E2A  91
7E2B  93                    .BYTE $93,$8D,$12,$92
7E2C  8D
7E2D  12
7E2E  92
7E2F  14                    .BYTE 20,0,0,0
7E30  00
7E31  00
7E32  00
7E33  00                    .BYTE 0,0,0,0
7E34  00
7E35  00
7E36  00
7E37  00                    .BYTE 0,0,0,0
7E38  00
7E39  00
7E3A  00
7E3B  00                    .BYTE 0,0,0,0
7E3C  00
7E3D  00
7E3E  00
7E3F              ; The following code is executed after
7E3F              ; each 'jiffy' interrupt. It tests for
7E3F              ; several 'special' keystrokes to allow
7E3F              ; enhanced screen editing, setting/clearing
7E3F              ; of control mode, and sending BRK's.
7E3F              ;
7E3F  A5 9E       KEYTST LDA NDX          ;any chars waiting?
7E41  F0 25              BEQ REXIT        ;no
7E43  AD 6F 02           LDA KEYD         ;next char in key buffer
7E46  A6 CD              LDX CRSPRT       ;in quote mode?
7E48  D0 16              BNE KEY05        ;yes, test rvs/stop
7E4A  A6 98              LDX SFST         ; or shift key up?
7E4C  F0 12              BEQ KEY05        ;yes, test rvs/stop
7E4E  A2 0C              LDX #NKYS        ;scan replace table
7E50  DD 9D 7E    KEY02  CMP KEYTAB,X     ;special key?
7E53  D0 08              BNE KEY04
7E55  BD A9 7E           LDA RPLTAB,X     ;change to alt. chr.
7E58  8D 6F 02           STA KEYD
7E5B  B0 0B              BCS REXIT
7E5D  CA          KEY04  DEX
7E5E  10 F0              BPL KEY02
7E60  C9 12       KEY05  CMP #$12         ;a RVS?
7E62  F0 07              BEQ CTLTGL       ;yes, toggle ctrl
7E64  C9 03              CMP #3           ;a STOP?
7E66  F0 17              BEQ KEYBRK       ;do break
7E68              REXIT                   ;done
7E68  4C 1C 7E           JMP IRQDON
7E6B              ; Set or clear control mode
7E6B              ;
7E6B  A9 22       CTLTGL LDA #$22
7E6D  A6 CD              LDX CRSPRT       ;in quote mode?
7E6F  F0 05              BEQ CTL100       ;no, enter it
7E71  8D 6F 02           STA KEYD         ;exit quotes
7E74  A9 92              LDA #$92         ;exit RVS
7E76  8D 70 02    CTL100 STA KEYD+1
7E79  A9 02              LDA #2
7E7B  85 9E              STA NDX          ;2 keys in buffer
7E7D  D0 E9              BNE REXIT
```

SMARTERM Listing *(continued)*

```
LOC   CODE        LINE
7E7F              ; Send BRK for as long as operator holds down 'STOP'.
7E7F              ;
7E7F  A9 00       KEYBRK LDA #0           ;clear buffer
7E81  85 9E              STA NDX
7E83  A5 9B              LDA $9B          ;STOP still down?
7E85  C9 EF              CMP #$EF
7E87  D0 0A              BNE KBKDON
7E89  AD F2 EF           LDA $EFF2
7E8C  09 04              ORA #4           ;set BRK
7E8E  8D F2 EF           STA $EFF2
7E91  D0 EC              BNE KEYBRK
7E93  AD F2 EF    KBKDON LDA $EFF2
7E96  29 FB              AND #$FB         ;clr BRK
7E98  8D F2 EF           STA $EFF2
7E9B  D0 CB              BNE REXIT
7E9D              ; Table of special keys to be replaced (when
7E9D              ; they are pressed while holding the 'SHIFT'
7E9D              ; key) by screen editing function characters
7E9D              ; not directly accessable on the CBM keyboard.
7E9D              ;
7E9D  1B          KEYTAB .BYT 27,'0123456789.' ;'ESC' & numeric pad
7E9E  30 31
7EA9  16          RPLTAB .BYT 22          ;ESC = erase to EOL
7EAA  95                 .BYT 149         ;'0' = insert line
7EAB  96                 .BYT 150         ;'1' = erase to beg.
7EAC  15                 .BYT 21          ;'2' = delete line
7EAD  99                 .BYT 153         ;'3' = scroll down
7EAE  11                 .BYT $11         ;'4' = crsr down
7EAF  0E                 .BYT $0E         ;'5' = enter text mode
7EB0  19                 .BYT 25          ;'6' = scroll up
7EB1  91                 .BYT $91         ;'7' = crsr up
7EB2  8E                 .BYT $8E         ;'8' = enter graphics mode
7EB3  0F                 .BYT $0F         ;'9' = set top of window
7EB4  8F                 .BYT $8F         ;'.' = set bot. of window
7EB5              NKYS   = RPLTAB-KEYTAB  ;# OF SPECIAL KEYS

7EB5  00 00       JIFFY  .WORD 00         ;Place to save IRQ vector

7EB7                     .END
```

BASIC DRIVER FOR SMARTERM

```
100 POKE 53,7*16:CLR:                          REM: protect code from basic
110 IF PEEK(32000)<>120 THEN LOAD "smrt5",8: REM: get code if necessary
120 cs$="[ 2 HOME][TEXT][ 24 DOWN]":    REM: crt format string
130 PRINT "[RVS] S M A R T E R M 5 . 0 "
140 INPUT "[DOWN]mode ([RVS]l[OFF]ine/[RVS]c[OFF]har.)  I[ 3 LEFT]";x$
150 md$="line-by-line":IF x$<>"l" THEN md$="char-by-char
160 POKE 13*16+3,-255*(x$<>"l"):               REM: set output mode flag
170 INPUT "[DOWN]display control chars (y/n)  n[ 3 LEFT]";x$
180 POKE 13*16+2,-255*(x$="y"):                REM: set cntrl char flag
190 cc$="invisible":IF x$="y" THEN cc$="visible
200 SYS 32000:                                 REM: turn on 6551
210 GOSUB 270:                                 REM: disp. modes bot. line
220 : REM: the following line could be replaced by auto-logon code
230 PRINT "[CLR]You should now attempt to logon.
240 SYS 32031:PRINT:                           REM: goto term. routine
250 IF st=19 THEN 130:                           REM: i.e. stop key pressed
260 PRINT "[RVS]Lost carrier.":END
270 PRINT cs$"[SET TOP][CLR][RVS]mode:  "md$;
280 PRINT TAB(30)"control chars:  "cc$cs$"[UP]" TAB(79)"[SET BOT][HOME]"
    :RETURN
```

Dialing the Networks

by Cliff Glennon

Essential steps for a MC6809-based home computer to communicate with the two major computer networks.

Have you ever come home without a newspaper and wished you had something to read? Are you tired of paying high prices for slow mail delivery? Subscribers to THE SOURCE and COMPUSERVE can get news, instant electronic mail, and a host of other valuable services delivered right to their home computers.

There are so many services offered by COMPUSERVE that a magazine is published to provide a convenient index to them. THE SOURCE sends out an executive manual that covers its services. I had the spelling in this article checked by THE SOURCE, and COMPUSERVE can give me the prices of stereo equipment.

I have heard it said that by the next decade a literate person will have to know computers to be able to communicate. This communication most likely will be over the telephone lines through computer services such as THE SOURCE or COMPUSERVE. If you want a taste of what it will be like to work at home and communicate to a large central computer system, it's all here. If you have a program that will not fit in your memory, you have access to all the memory and disk operating systems you can handle over your telephone.

The Modem

Modem is a contraction for modulator-demodulator. Although I don't have one, the Originate-Answer type of modem is probably best. This

Dialing requires:

6809-based microcomputer

***Table 1:* Summary of the MC6850 ACIA Command Register**

bits 0-1) 000000bb
Divides the system clock to provide output baud rates.
00 - Divide by 1
01 - Divide by 16
10 - Divide by 64
11 - Reset the ACIA
The SWTPC MP-S2 is set up to use a divide by 16 when 300 baud is selected on the interface jumpers.

bits 2-4) 000bbb00
Word Length, Parity, Stop Bit Selections
000 - 7-bit word, even parity, 2 stop bits
001 - 7-bit word, odd parity, 2 stop bits
010 - 7-bit word, even parity, 1 stop bit
011 - 7-bit word, odd parity, 1 stop bit
100 - 8-bit word, no parity, 2 stop bits
101 - 8-bit word, no parity, 1 stop bit
110 - 8-bit word, even parity, 1 stop bit
111 - 8-bit word, odd parity, 1 stop bit
I access both COMPUSERVE and THE SOURCE with a 7-bit word, even parity, 1 stop bit, %00001000 or 010.

bits 5-6) 0bb00000
Controls RTS output (pin 5), Break Transmission, Transmitter Interrupts
00 - RTS = 0, inhibits transmitter interrupt
01 - RTS = 0, enables transmitter interrupt
10 - RTS = 1, inhibits transmitter interrupt
11 - RTS = 0, inhibits transmitter interrupt, transmits Break
I do not use these features.

bit 7) b0000000
Controls Interrupts to the 6809 Processor
1 - enables interrupts when a letter is received
0 - disables interrupts when a letter is received
I have used this feature in the past, but my Disk Operating System uses the interrupt vector and I hesitate to share that vector when I am using disk reads and writes. Also I found that using interrupts prevents control characters from being sent to the Services (e.g., a break or Control-P) by assigning a priority to incoming letters.

means that you can be the one to initiate the call (Originate) or that your computer can be called by another computer (Answer). I have an Originate-only modem, and this is sufficient to connect to the computer services.

My modem is a direct-connect, which means there is no acoustic coupler to add problems to the communications channel. I see no need to convert the electronic signals from the computer to sound, and convert the sound back to electronic signals to send over the phone lines. In addition, acoustic couplers are made for round phone speakers, and my phone handset is square. The phone company installed the USOC RJ-11C jack required by the modem. This jack, as it turns out, is also required by my telephone answering machine and enables me to plug or unplug phone equipment easily.

The two Services require at least a 300-baud rate:

baud = (approx.)10*characters/second

but also provide 1200-baud service. The future undoubtedly will be with the faster baud rates and a modem that could operate at such speeds would be an advantage.

The Cable

If you construct your own modem-computer connector, you must translate the modem manufacturer's terms to the computer manufacturer's terms. The name RS-232 is code for a loose agreement "standard" for connectors that original equipment manufacturers (OEMs) can use to attach their devices to a variety of computers. As long as a device follows the RS-232 standard, I can attach it to my SWTPC S09 computer. Here are the modem-to-computer conversions:

Modem Pin	Line Description	Computer (SWTPC MP-S2)Pin
(1)	Protective Ground	(1)
(2)	Transmitted Data	(3) < **note well
(3)	Received Data	(2)
(5)	Clear to Send	Not connected
(6)	Data Set Ready	(20) Clear to Send
(7)	Signal Ground	(7)
(8)	Data Carrier Detector	(12) SDCD

The first thing to notice in the list is that lines 2 and 3 are reversed in the two machines. This is a standard configuration and should be found in all modem-computer connections. The Data Carrier Detector line does not have to be connected for the MP-S2 interface to work. A very careful reading of the SWTPC documentation discloses that pin 20, the Clear-to-Send pin, should be connected to "the buffer full or data terminal ready line." All in all, only five lines need to be implemented.

The cheapest cable is ribbon cable. But a major disadvantage is that the signals on this cable radiate to interfere with any television sets in your house. If you live in an apartment, ribbon cable is out; you should have a cable custom made with the lines twisted and mylar shielded. Another alternative is to adapt an unused shielded cable.

DB-25 is the name for the 25 pin connectors used with RS-232 interfaces. They are male and female to indicate whether they are plugs or sockets. If you order the cable made, be sure you understand how the manufacturer wants the gender of the DB-25 connector specified. Serial interfaces usually require male DB-25 connectors; parallel interfaces need female connectors. Cable and connectors can be purchased from computer stores or hobby mail-order houses.

Attaching wires to the connectors is easy. A low-wattage soldering iron and 60/40 rosin-core solder is all that is necessary. A short length of heat-

***Table 2:* The MC6850 ACIA Status Register**

bit 0) 0000000b
- 0 - Receiver Data Register empty
- 1 - Receiver Data Register full
  A character has been received and can be read from the Data Register

bit 1) 000000b0
- 0 - Transmitter Data Register full
  ** Note opposite meanings from bit 0
- 1 - Transmitter Data Register empty
  A character can now be sent

bit 2) 00000b00
- 0 - Data Carrier Detect is present
- 1 - Loss of Data Carrier
  If this line is connected

bit 3) 0000b000
- 0 - Clear to Send signal is detected
- 1 - No Clear to Send

**** Note: this line must be connected for the 6850 to operate. If this line is high ($08 in the Status Register), no data can be transmitted. This is pin 20 on the MP-S2 connector

bit 4) 000b0000
- 0 - No Framing Error
- 1 - Framing Error
  Faulty character synchronization

bit 5) 00b00000
- 0 - No Overrun
- 1 - Overrun
  More than one character was received before one was read

bit 6) 0b000000
- 0 - No Parity Error
- 1 - Parity in the received character is incorrect

bit 7) b0000000
- 0 - Any interrupts enabled in the Control register
- 1 - Can also be read as output in this bit

shrinkable tubing is slipped over the wire before soldering. After the solder connection is made, this tubing is pulled down over the connection and shrunk to a tight fit by heat from the iron; or you can use plastic electrical tape if you prefer. A VOM can be used to check if there are any invisible breaks in the wire, if the right pin is connected, or if there is a short between wires. An inexpensive VOM is sufficient, because only resistance measurements are needed.

The next step is to write the program that allows the computer to talk on the telephone. A preliminary procedure is to study the device used in the computer interface to find out the commands it needs to operate. The device in my system is the Motorola MC6850 ACIA, or Asynchronous Communications Interface Adapter (I am curious to see what the spelling checker does with that!). To send a command to the 6850, a value (such as $03) is placed in the Control Register. For example:

```
LDA $03       Load accumulator A with
              the 6850 Reset value
STA $E040     Control/Status Register
              address in my system
```

The commands are coded to fit into an 8-bit byte (see table 1). If table 1 seems complicated, remember that all you have to do is select one option in each of the categories to fit your needs and the 6850 does the rest! Thus, COMPUSERVE asks for a 7-bit ASCII word, even parity, one start, and one stop bit. All this is done with a $09 or %00001001. After sending this command to the 6850, all data sent out by the computer to the modem conforms to this requirement, and data received is checked to see if it matches as well. Characters are transmitted and received simultaneously.

THE SOURCE looks for an 8-bit ASCII word, no parity, one stop bit. This is obtained with a $15 or %00010101. I am able to connect on my SWTPC 6800 system using this command; but my 6809 system balks at this code and talks only on the $09 code. Customer service at THE SOURCE told me that a 7-bit word could be used to communicate, but that an 8-bit word is required in their "local mode," which, I guess, is dialing from Washington D.C. My motto in this case is "what works, works," but I am sure I will have to find the source of the trouble someday.

Both services require full-duplex operation, which means the service will echo a character sent by your equipment back to you. Note that you do not have to echo a character back to the service. Full-duplex operation is assumed in the attached program.

The computer processor is processing data at a megahertz-cycle clip, and the ACIA modem is running at only 300 baud, so a means must be provided to see if the slow pair is ready for another letter. This is provided by the status register, which tells us whether or not a letter has come in, or what some of the problems in the reception are. On my system, this register is read by an

```
LDA $E040  Reads the ACIA Status
           Register
```

The status register is summarized in table 2.

It is necessary only to check bits 0 and 1 for normal communications. If a parity option has been selected such as a $05: 7-bit word; odd parity; and divide by 16, and the parity status register bit number 6 is not checked by a statement such as

```
LDA $E040 Read status
BITA $40 Check parity bit
```

then you are sending characters out with a parity bit set, but your own system is ignoring any parity bits received.

The final piece of information is how to read and write to the ACIA. The required statement is:

```
LDA $E041 Read Data Register
                 or
STA $E041 Write to Data Register
```

The Program

After loading the progam, it prompts for a letter to begin initializing the ACIA. Enter any letter to start. Dial the computer service at this time and follow the sign-on procedures detailed in their instructions. To record any information, type a '~' or $7E. To stop recording, enter another '~' or $7E. To transmit a text file, type a '{' or $7B. Do not be alarmed if the characters echoed back by the service during the transmission of a text file do not agree with the characters that are being sent. More than likely, when you review the file in the service's memory, it will agree with what you intended to send. But (and there must always be a "but") a poor telephone line or static on the telephone line may garble the best transmissions. You must not touch the keyboard during transmission because this will end the transmission. Use this method to end the transmission, however, if the service sends out trouble messages such as

```
> illegal command
```

If the file TODAY.TXT is inadvertently closed, you can exit and restart the MODEM program without losing the telephone connection. Exit from the program by typing a '}' or $7D.

Problems

To locate a problem you must first isolate it by eliminating any areas of the connection that are not (or should not be) involved. Generally I assume anything that I have done is wrong, even though I know that I am right beyond a shadow of a doubt. This attitude has solved most of my problems quickly. Any manufactured and tested part is probably not the source of the problem.

Of course, I hope you do not have any problems with the program as it is printed here. It is designed for exchanging text messages. Binary transmissions, such as machine-language program exchanges, would require that parity and framing errors be detected. Error-correcting codes would also have to be employed to achieve 100% accuracy.

To paraphrase Professor James Burke in his CONNECTIONS series: The inventions that will probably be the most important are the ones that will improve communications.

P.S. The SPELL program caught my "propriatary" and pointed out the correct "proprietary." "Asynchronous" passed by "synchronization" was unknown. The SPELL program also listed all the text formatter commands that are imbedded in the text (such as centering, etc.) as unknown words.

You may contact the author at 3395 Nostrand Ave., Brooklyn, NY 11229.

Glennon Listing

```
                        NAM MODEM
            .
                        PROC
            .
            .   FLEX9 ENTRY POINTS
            .
D4D6        FMS         EQU $D406      FILE MANAGEMENT
CD24        PCRLF       EQU $CD24      CARRIAGE RETURN
            .                          & LINE FEED
C01B        INBUFF      EQU $CD1B      INPUT INTO LINE BUFFER
COBO        LINBUF      EQU $COBO      LINE BUFFER ADORESS
CC14        BUFPOINT    EQU $CC14      LINE BUFFER POINTER
CD2D        GETFIL      EQU $C020      GET FILE SPECIFICATION
COIE        PSTRNG      EQU $CDIE      PRINT STRING
CD15        GETCHR      EQU $COI5      INPUT 1 CHARACTER
CD3F        RPTERR      EQU $C03F      REPORT ERRORS
D403        FMSCLS      EQU $D403      CLOSE OPEN FILES
CD03        WARMS       EQU $CD03      EXIT TO FLEX9
            .
            .   ACIA ADORESSES
            .
E040        CNTRL       EQU $E040      CONTROL/STATUS REGISTER
E041        DATA        EQU $E04I      DATA REGISTER
            .
            .   TERMINAL ACIA ADDRESSES
            .
E004        TCNTRL      EQU $E004      CDNTROL/STATUS REGISTER
E005        TOATA       EQU $E005      OATA REGISTER
            .
            .   PROGRAM ENTRY POINT
            .
0000 BE 02 B4      START:   LDX #FCB       THE FILE CONTROL
                   .        BLOCK (FCB) IS A 320 BYTE BLOCK USED BY
                   .        FLEX9 TO CONTROL OISK INPUT/DUTPUT.
                   .
0003 10 8E 02 72            LDY #FILSPEC   A CONVENIENT
                   .        WAY TO INITIALIZE A FCB FOR USE BY
                   .        FLEX9.
                   .
0007 C6 OF                  LDB #15        15 LETTERS
0009 A6 A0         STLOOP   LDA D,Y+       ARE WRITTEN
000B A7 80                  STA 0,X+       TO THE BLOCK.
OD0D 5A                     OECB
000E 2A F9                  BPL STLOOP
                   .
00I0 8E 02 84               LDX #FCB       RESET ACCX
00I3 BD D4 06               JSR FMS        OPEN THE FILE
0016 10 26 0I 55            BNE ERROR      ERROR TRAP
                   .
001A 8E 01 96               LDX #PROMPT    PAUSE BEFORE
001D BD CD 1E               JSR PSTRNG     INITIALIZING ACIA
0020 BD CD 15               JSR GETCHR     INPUT CHARACTER
                   .
0023 86 03         RESET    LOA #3         RESET ACIA
0025 B7 E0 40               STA CNTRL
                   .
0028 86 09                  LDA #$09
002A B7 E0 40               STA CNTRL
                   .
002D 7F 02 7I               CLR FLAG       OISK WRITE FLAG
DD30 8E 01 8B               LDX #READMSG   READY MESSAGE
D033 BD CD IE               JSR PSTRNG     WRITE TO TERMINAL
                   .
                   .   INPUT A LETTER FROM THE TERMINAL
                   .
0036 B6 ED D4      TERM     LDA TCNTRL     CHECK IF TERMINAL
0039 85 01                  BITA #$01      KEY IS OEPRESSEO
                   .
003B 27 33                  BEQ PORT       NO. GO SEE IF
                   .           MODEM HAS ANYTHING.
                   .
DD3D B6 ED 05               LDA TDATA      YES. GET CHAR.
                   .
0040 BI 7E                  CMPA #'='      TOGGLE OISK WRITE?
0042 26 IB                  BNE CKNO       NO. AND SKIP
D044 73 02 71               CDM FLAG       YES. FLIP SWITCH
DD47 7D 02 7I               TST FLAG       OECIOE ON MESSAGE
004A 27 D5                  BEQ MSGOFF     RECORD OFF
D04C BE 02 37               LOX #MSGON     RECORD ON
004F 20 03                  BRA MSG666     PUT ON TERMINAL
0051 8E D2 5D      MSGOFF   LDX #DFFMSG    RECORD OFF
0054 BD CD IE      MSG666   JSR PSTRNG     REPORT IT
0057 8E 02 84               LOX #FCB       RESTORE POINTER
DD5A 20 14                  BRA PORT       GD CHECK ON PORT.
                   .
005C B1 7D         CKNO     CMPA #'†'      GIVE UP?
005E 10 27 01 D0            BEQ OOENO      YES.
0062 8I 7B                  CMPA #'§'      TRANSMIT A FILE?
0064 27 4D                  BEQ FILETRANS  YES.
                   .
0066 F6 ED 4D      DUTCH    LDB CNTRL      TRANSMITTER REAOY?
0069 C5 02                  BITB #2
006B 27 F9                  BEQ OUTCH      NO. WAIT UNTIL READY
                   .
0060 B7 E0 41               STA OATA       READY, SENO OATA
                   .
                   .   INPUT A LETTER FROM THE MOOEM.
                   .
0070 F6 E0 40      PORT     LDB CNTRL      DID ANYTHING
0073 C5 01                  BITB #1        COME IN?
0075 27 BF                  BEQ TERM       NO. GO CHECK TERMINAL.
                   .
0077 B6 E0 41               LDA DATA       GET LETTER
                   .
                   .  THE FOLLDWING SCREEN IS NECESSARY TO
                   .  PREVENT A STRAY MISREAO CHARACTER FROM
                   .  ACTIVATING ANY OF THE 150 FUNCTIONS ON
                   .  THE SWTPC CT-B2
                   .
007A 81 OD                  CMPA #$0D      CARRIAGE RETURN?
007C 27 16                  BEQ PORT1      OK
                   .
007E B1 0A                  CMPA #$0A      LINE FEED?
00B0 27 12                  BEQ PORT1      OK
                   .
00B2 B1 08                  CMPA #$0B      BACKSPACE?
00B4 27 0E                  BEQ PORTI      OK
                   .
00B6 B1 07                  CMPA #$07      CHIMES?
00BB 27 0A                  BEQ PORT1      OK
                   .
00BA B1 20                  CMPA #$20      OTHER CONTROLS?
00BC 25 04                  BLD WHAT       NOT OK
00BE B1 7E                  CMPA #$7E      NOT ASCII?
0090 23 02                  BLS PORTI      NOT OK
0D92 B6 7C         WHAT     LDA #'¶'       A SYMBOL TO INDICATE
                   .         THAT A BAD CHARACTER WAS RECEIVED.
                   .
0094 7D 02 7I      PORTI    TST FLAG       WRITE LETTER
                   .                          TO DISK?
0097 27 0E                  BEQ PORT2      NO. SKIP
                   .
0099 34 02                  PSHS A         SAVE ACCA
009B 8E 02 B4               LDX #FCB       WRITE LETTER
009E B0 04 06               JSR FMS        TO OISK.
00AI 10 26 00 CA            BNE ERROR      ERROR TRAP
00A5 35 02                  PULS A         RESTORE ACCA
                   .
00A7 F6 E0 04      PORT2    LOB TCNTRL     TERM READY?
00AA C5 02                  BITB #$02
00AC 27 F9                  BEQ PORT2      NO. WAIT
                   .
00AE B7 E0 05               STA TDATA      SEND CHARACTER
D0B1 20 83                  BRA TERM       AND GO CHECK TERMINAL.
                   .
                   .   TRANSMIT A DISK FILE
                   .
00B3 8E 0I B4      FILETRANS    LDX #TRANSMSG  PROMPT FOR FILE NAME
00B6 B0 CD IE               JSR PSTRNG     PRINT PROMPT
00B9 8E C0 80               LDX #$C080     LINE BUFFER ADORESS
00BC BF CC 14               STX $CC14      BUFFER POINTER
00BF B0 C0 1B               JSR INBUFF     INPUT FILE NAME
00C2 8E C0 80               LDX #$C080     LINE BUFFER ADORESS
00C5 BF CC 14               STX $CC14      BUFFER POINTER
00C8 8E 03 C4               LDX #FCB2      NEW FILE CONTROL
                                           BLOCK
00CB BD CD 2D               JSR GETFIL     ENTERS THE FILE
                   .                       SPECIFICATION INTO THE
                   .                       FILE CONTROL BLOCK
0OCE 86 01                  LOA #$01       OPEN FOR READ CODE
00D0 A7 84                  STA 0,X        SET FCB
D0D2 B0 04 06               JSR FMS        OPENS THE FILE.
00D5 26 6D                  BNE ERROR6     ERROR TRAP
                   .
                   .   REAO EACH CHARACTER FROM THE FILE
                   .
00D7 8E 03 C4      READ_CHAR    LOX #FCB2      POINTS TO FCB
00DA BD D4 D6               JSR FMS        LOAOS ACCA WITH CHAR.
D0DD 26 65                  BNE ERROR6     LOOKS FOR ENO OF FILE
                   .
                   .   ELIMINATE MORE THAN ONE CARRIAGE RETURN
                   .   TO PREVENT RETURNING TO COMMAND MODE IN
                   .   THE SOURCE.
                   .
00DF 8I 0D                  CMPA #$0D      CARRIAGE RETURN?
00E1 26 0C                  BNE NOT_CR     NO.
00E3 70 02 70               TST CRFLAG     WAS LAST A C.R.?
00E6 26 EF                  BNE READ_CHAR  YES SKIP THIS
00E8 C6 FF                  LOB #$FF       NO. BUT SKIP
00EA F7 02 70               STB CRFLAG     ALL SUBSE-
00ED 20 03                  BRA WAIT22     QUENT C.RS.
00EF 7F 02 70      NOT_CR   CLR CRFLAG     CLEAR FLAG
                   .
00F2 F6 E0 40      WAIT22   LDB CNTRL      CHECK FOR EMPTY
00F5 C5 02                  BITB #$02      TRANSMITTER.
00F7 27 F9                  BEQ WAIT22     NOT REAOY.
00F9 B7 E0 41               STA OATA       SENO CHARACTER
DDFC F6 E0 4D      WAIT30   LDB CNTRL      WAIT FOR ECHO
00FF C5 0I                  BITB #$01      FROM HOST
0I01 26 I4                  BNE WAIT OVER
DI03 F6 ED 04               LDB TCNTRL     ALLOW EXIT FROM
0I06 C5 0I                  BITB #$0I      LOOP
0I08 27 F2                  BEQ WAIT30
010A 26 73                  BNE ERR66      TRAP ERROR
010C 8E 01 EE               LDX #TRANSINT  TRANSMISSION-
```

Glennon Listing *(continued)*

```
D10F  BD CD 1E              JSR PSTRNG      INTERRUPTED.
D112  8E D3 C4              LDX #FCB2
0115  2D 2D                 BRA ERROR6
                    *
0117  B6 ED 41     WAIT_DVER     LDA DATA        PICK UP
                                                 RETURNED CHAR
011A  81 0D                 CMPA #$0D       ALLOW A CARRIAGE RET.
011C  27 0C                 BEQ PASS_OVER
011E  81 0A                 CMPA #$0A       ALLOW A LINE FEED
0120  27 08                 BEQ PASS_OVER
0122  81 2D                 CMPA #$20       SCREEN IT
0124  25 10                 BLD BAD_ECHO
D126  81 7E                 CMPA #$7E       ASCII?
0128  22 0C                 BHI BAD_ECHO
012A  F6 ED 04     PASS_DVER     LDB TCNTRL      TERMINAL READY?
012D  C5 D2                 BITB #$02
012F  27 F9                 BEQ PASS_OVER   NOT YET.
0131  B7 E0 05              STA TDATA       SEND TO TERMINAL
0134  20 A1                 BRA READ_CHAR   GET NEXT CHARACTER.
                    *
D136  86 7C        BAD_ECHO LDA #'¶'        BAO ECHO INOICATOR
0138  F6 E0 04     BAOBAD   LOB TCNTRL      TERMINAL READY?
013B  C5 02                 BITB #$02
013D  27 F9                 BEQ BAOBAD      NDT YET
013F  B7 E0 05              STA TDATA       SENO TO TERMINAL
0142  20 93                 BRA REAO_CHAR   TRY NEXT CHARACTER.
                    *
0144  A6 01        ERROR6   LDA 1,X         ERROR CODE
0146  81 08                 CMPA #$08       ENO OF FILE
D148  27 03                 BEQ CLOSE_SHOP   YES THE END.
014A  BD CD 3F              JSR RPTERR      REPORT OTHER ERROR
014D  86 D4        CLOSE_SHOP    LDA #$04        CLOSE FILE CDDE
014F  A7 84                 STA 0,X
0151  BD D4 06              JSR FMS         CLOSE FILE
0154  26 29                 BNE ERR66
0156  8E 02 15              LDX #TRANSCOMP   END TRANSMISSION
0159  BD CD 1E              JSR PSTRNG      PRINT MSG
015C  8E 02 84              LDX #FCB        RESTORE POINTER
015F  16 FE D4              BRA TERM        RETURN TO MAIN
                    *                       PROGRAM LDDP.
                    *
                    *   EXIT PROGRAM
                    *
0162  8E 02 84     DOEND    LDX #FCB        TODAYS RECORO
0165  86 04                 LDA #$4         CLOSE FILE CODE
0167  A7 84                 STA 0,X         CLOSE THE FILE
D169  BD D4 06              JSR FMS
016C  7E CD D3              JMP WARMS       AND RETURN TO FLEX9
                    *
                    *   DISK OPERATION ERRORS
                    *
                    *   IF THE FILE TODAY.TXT EXISTS IT
                    *      MUST BE DELETED.
016F  A6 01        ERROR    LDA 1,X         GET ERROR COOE
0171  81 03                 CMPA #3         FILE EXISTS?
0173  26 0A                 BNE ERR66       NO. REAL TROUBLE
                    *
0175  86 0C                 LOA #12         DELETE FILE
D177  A7 84                 STA 0,X
0179  B0 D4 06              JSR FMS
017C  16 FE 81              BRA START       ANO TRY AGAIN
                    *
017F  BD CD 3F     ERR66    JSR RPTERR      REPORT ERROR
0182  BD D4 03              JSR FMSCLS      CLOSE ALL FILES
0185  73 02 71              COM FLAG        CLEAR WRITE FLAG
0188  16 FE AB              BRA TERM        CONTINUE TO RECEIVE
018B  52 45 41 44  READMSG  FCC /READY/
0190  DA 0D 00 00           FCB $0A,$0D,D,D,0,4
0196  54 59 50 45  PROMPT   FCC /TYPE ANY LETTER TO START/
01AE  0A 0D 00 00           FCB $0A,$0D,0,0,0,4
01B4  45 4E 54 45  TRANSMSG FCC /ENTER FILE SPECIFICATION FOR FILE
                                 TO BE TRANSMITTEO./
01E8  0A 0D 00 00           FCB $DA,$0D,D,0,0,4
01EE  49 4E 54 45  TRANSINT FCC /INTERRUPT RECEIVED FROM
                                           TERMINAL./
020F  0A 0D 00 D0           FCB $0A,$0D,0,0,0,4
0215  54 52 41 4E  TRANSCOMP     FCC /TRANSMISSION FILE IS CLDSED./
0231  0A 0D 00 00           FCB $0A,$00,0,0,0,4
0237  20 20 20 2B  MSGON    FCC /   ++++ RECDRDING ON ++++/
0250  20 20 20 2A  OFFMSG   FCC /   **** RECORDING OFF ****/
026A  0A 0D 0D 00           FCB $0A,$00,0,0,0,4
0270               CRFLAG   RMB I           CARRIAGE RETURN FLAG
0271               FLAG     RMB 1
0272  D2 00 00 00  FILSPEC  FCB 2,0,D,0,"TODAY",0,0,0,"TXT",0,0,D
0284               FCB      RMB 32D         RECORD FILE
D3C4               FCB2     RMB 32D         TRANSMIT FILE
                            END
0D00                        END START
-- NO ERRORS THIS ASSEMBLY.
```

MICRO™

A Home-Built Communications Interface

by John Steiner

Circuitry and techniques to construct a communications interface. With modifications could be converted to a telephone modem. Simple, reliable, and inexpensive design.

Communication between computers is rapidly becoming a common-place occurrence. More and more people are involved with electronic mail, time sharing, and data base activities. Mechanical radio teletype systems are being replaced by modern computer technology, and the Baudot code is being supplanted by ASCII. This article describes the construction and connection of a radio teletype modem. Techniques found here can be applied to any digital data communications application.

The modem can act as an interface with any serial RS-232-C device, but this article describes the process used to connect it specifically to the TRS-80 Color Computer. In this case the equipment being interfaced is an amateur radio transceiver; with some changes it would be possible to convert this device to a telephone-type modem.

The TRS-80 Color Computer has proven to be an excellent communications terminal. It is inexpensive, easily programmed, and includes an RS-232 output connection. CoCo is well shielded from external sources of radio frequency interference and causes little of its own. After reading several articles in various periodicals and books, Ken Christiansen (W0CZ) and I decided we would like to experiment with radio teletype (RTTY). We selected a basic demodulator design from the National Semiconductor Data Book. The modulator is modified from a basic circuit by Rodney Colton (WA1SXW) in an article in *QST* magazine, September 1981.

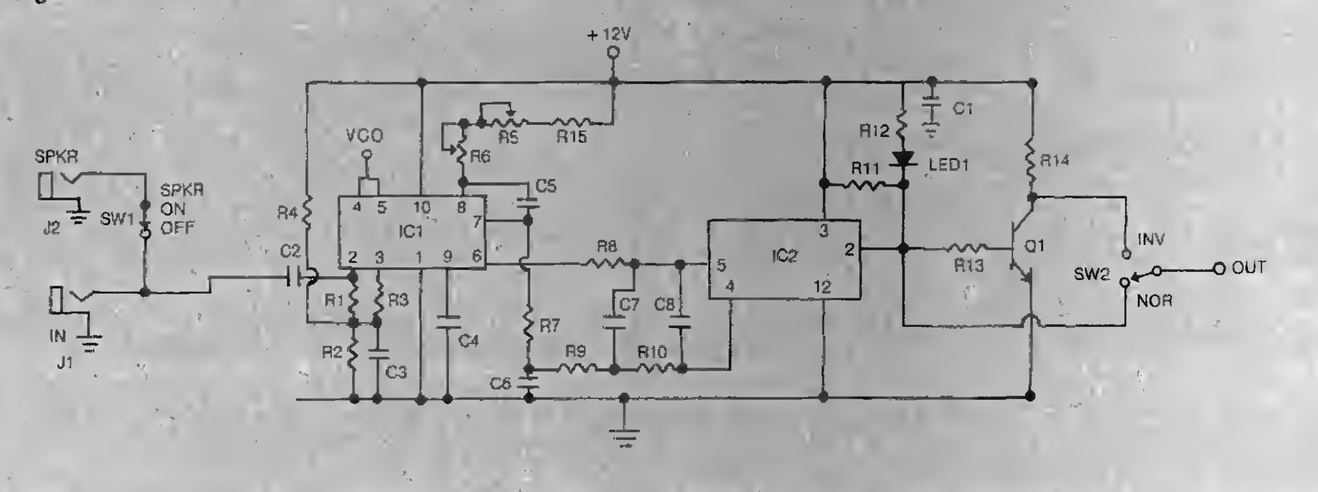
In our research, we found several interesting articles and books. The bibliography lists those that were especially helpful to us in learning about RTTY. Ken and I were interested in communicating *via* two meters, so frequency offsets were designed around the VHF convention of 170 Hz frequency shift. The mark frequency is 2125 Hz and space is 2295 Hz. Also included is a voltage-regulator circuit that ensures stability of operation of the PLL circuits. The modems have been used occasionally on the high-frequency bands, but a lack of filtering hampers their performance. One of these units has been used with excellent results with audio filtering preceding the demodulator.

Total cost for all components, if purchased new, should be $25 to $35, depending on final configuration and cabinet. The modem is designed to be powered from a 14-volt or higher DC source. A simple supply can be built for under $20, if one is not available. I use an inexpensive CB radio-power supply.

Demodulator Circuit

The simple FSK demodulator uses a 565 phase-lock loop IC and is a modified circuit originally found in the NS data book. The circuit has excellent stability and has worked flawlessly for several months now. IC 1 (see figure 1) is the PLL. The circuit is adjusted with R5 and R6 to be between the high (mark) and low (space) frequencies.

Figure 1



Mark and space audio tones input to C2 cause the PLL output (pin 7) to be higher or lower than a reference voltage (pin 6). IC2, a comparator, compares the voltages and responds with a logic zero or logic one at the output (pin 2).

A few features have been added to the circuit to make it more versatile. R5 is mounted on the front panel and is a fine-frequency adjustment used to tune the PLL precisely to the input frequency. LED1 allows a visual indication of the data input. In practice, R5 is adjusted until the LED blinks with the changing data. Once the LED is blinking, you merely adjust for intelligible data on the CRT. Incorrect adjustment of R5 causes the LED to remain either on or off. Q1 is an inverter that reverses the state of the output logic, ensuring compatibility with any transmission standard. J2 is provided to connect an external speaker, making it easy to use the earphone jack on the transceiver and allowing you to monitor the incoming signal. SW1 can turn off the speaker once communication is established.

To adjust the demodulator, place a 2210 Hz signal on the input. Set R5 to midrange, then adjust R6 until the LED changes state as you turn the potentiometer back and forth. Check to see that the LED changes state as you bring the audio frequency between mark and space frequencies. If you cannot adjust the output within range, you may have to change R15 slightly.

Modulator Circuit

The modulator circuit uses a 566 PLL IC as a frequency generator. The input to the modulator is serial binary data from the computer. A high causes the mark frequency to be sent, and a low causes the space frequency to be sent. Q1 is an inverter that allows the logic to be inverted. If you have software that can complement the output data, these associated components can be removed. Q2 is a switch that is used to change output frequency. When the modulator is receiving a high, this switch is on. Frequency is determined by the specific adjustments of R7 and R8 and the voltage divider of R9 through R12. When the input goes low, Q2 shuts off, switching R7 and R8 out of the circuit. During space, R10 and the associated divider resistors determine the output frequency.

To adjust the circuit, ground the input. This switches R7 and R8 out of the circuit. Adjust R10 for the space frequency at the output as measured on a frequency counter. Put +5 volts on the input and adjust R8 to midrange and tune R7 until the output is at the mark frequency. Ground the input again and recheck space frequency. You will notice some interaction between the mark and space controls. Only slight adjustments will be required. As with the demodulator, you may have to change the value of R9 slightly if you cannot get the potentiometers within range. The entire process of adjusting the modem takes much less time to do than it does to describe!

SW2, a tone on/off switch, has been included to kill the tone without actually powering down the modem. As the unit warms up, it drifts very slightly. Let it run for a few minutes before making adjustments. Any drift in the demodulator is taken care of easily with the front panel control. Once warm, it is completely stable. We have had no long-term drift problems with the circuit.

Power Supply Regulator

The modem has a regulator circuit that helps stabilize the PLL circuits.

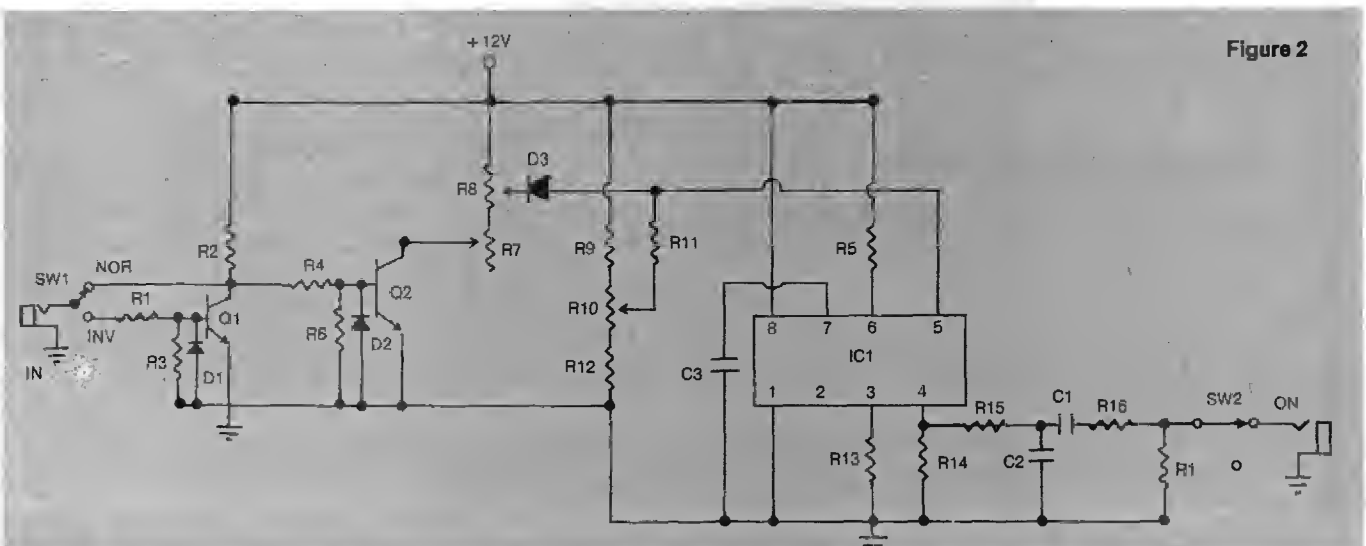


Figure 2

The heart of the circuit is a three-terminal IC — an LM 317 adjustable positive regulator. The circuit must have at least two volts more at the input than required at the output to retain regulation. The IC should be heat-sinked if you apply a very high input voltage. My regulator circuit gets its power from a 15-volt supply and does not run warm even without a heat sink. A power switch is included so that the main power supply can be left on for other purposes.

To adjust the circuit, connect a voltmeter to the output and adjust R1 until the meter reads 12 volts. Be sure to adjust the power supply output voltage before attempting to adjust the modem.

Construction

None of the circuits are critical, and they can be wired on printed circuit or perf board as desired. We have had three units constructed using the same basic circuit; even though the layouts have been totally different, each has worked without any problems for several months. You should use a metal cabinet if you plan to run the unit in high RF fields. We have not noticd any particular RFI problems with our units. Jacks and cable connectors that match the appropriate connectors on the transmitting device are required.

Interfacing the Modem

The connection between the Color Computer and modem is through the RS-232 jack marked SERIAL I/O on the rear panel of the computer. The easiest way to obtain the required four-pin DIN plug is to order the Radio Shack printer cable. If you cut it exactly in two, you will have two four-pin cables that can be used as I/O connections. The cable has color-coded conductors that are connected as follows:

Red to ground of modem
Green to output of demodulator
White to input of modulator
Yellow to positive voltage

Connection to the transmitter is *via* the audio output or external speaker jack. This connection goes between ground and the demodulator input. The modulator output connects between ground and the microphone or auxiliary input jack on the transceiver. In my particular installation, I ordered an external microphone for the handi-talkie, and installed a mini-stereo jack in it since I didn't want to drill into the case. As an added convenience, I connected the extra conductor in the stereo jack to the PTT line inside the microphone. This line is controlled by a switch on the modem marked XMIT, and allows me to remain in transmit without holding in the PTT switch.

When Ken and I completed the construction of the two modems, the only available software we knew about was Radio Shack's VIDEOTEX terminal program. This machine-language program operates at 300 baud ASCII with even parity protocol. Ken and I were assured of private transmissions as we were the only RTTY stations in the area with 300-baud capability. The modem operates at this speed with no problems, under normal two-meter reception conditions.

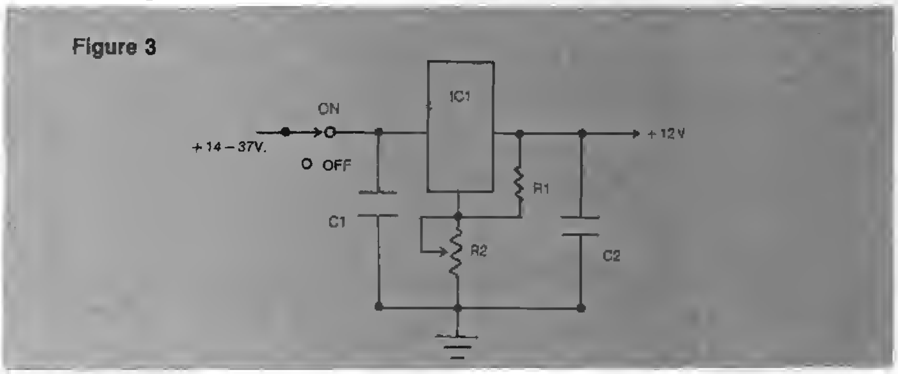
One evening I heard from a friend who spends much time on RTTY. He had just finished a contact with a station that was using a TRS-80C on 60 WPM Baudot, the standard used mostly on HF. Bill (W0LHS) told me that a radio ham was communicating with several individuals, all with color computers. He told of sending programs back and forth between terminals and informed me that the software they were using was called RTTYCW, written by K6AEP. Coincidentally Ken had just sent for a RTTY program he read about. His order to Clay Abrams Software was the same program — RTTYCW. It is capable of 60, 75, 100, and 110 WPM Baudot, as well as 50, 75, 100, 150, and 300 baud ASCII. The program will also send and receive morse code at 1 to 99 words per minute.

There are four message buffers and 12K transmit and receive buffers in a 32K CoCo. If you have a 16K machine, you are limited to a buffer size of about 4K. The transmit buffers can be loaded *via* tape, and all buffers can be saved to tape for loading at start-up time.

By loading a program saved in ASCII format into the transmit buffer, you can transmit that program to a receiver where it can be saved to tape. Then you can load the tape into the computer at a later time and resave it in standard format. If you want a hard copy of the text, all buffers can be sent to the printer. In short, I cannot say enough about the quality and capability of this software. It has all of the features I wanted when I thought of writing my own program.

The Color Computer is easy to interface, and the simple modem circuit has provided me with many hours of fun and education. The easy-to-adjust circuit can be built in just a few hours at little expense. If you have any questions or problems with construction, you may contact me at the address below, or on the Color Computer NET. This net meets at 2000 hours UTC Sundays on 14.343 Mhz, and I try to check in regularly. If you write, please enclose a stamped, self-addressed envelope for a reply.

Figure 3



Bibliography

1. ARPL Staff, "Radio Amateurs Handbook," American Radio Relay League, 1982.
2. Carr, Joseph J., "The UART," *Computers and Programming*, September 1981.
3. Colton, Rondey, "A PLL Demodulator and Modulator," *QST Magazine*, September 1981.
4. DeJong, Marvin L., "Morse Code Send/Receive Program," *Best of MICRO, Volume 3*.
5. *Henry, George W., Jr., "ASCII Baudot and Radio Amateur," QST Magazine*, September 1980.
6. Rouleau and Hodgson, *Packet Radio*, Tab Books, 1981.

You may contact John Steiner at ARS WB0NFX, 508 Fourth Ave. NW, Riverside, ND 58078.

PET-to-PET Communications

by F. Arthur Cochrane

This article describes a machine-language program to transfer an array from one PET to another over the User Port.

I have developed a method to communicate data between two Commodore PETs. Two PETs (PET A and PET B) are needed for on-line data collection and simultaneous graphic display and real-time monitoring of a chemical separations process. The tasks for PET A are instrument set-up, data collection, and data storage on disk. Tasks for PET B are graphic display, and reading and storing information on disk. The data for each transfer between PETs are limited to 14 floating-point values. For this application communication was necessary in only one direction — from PET A to PET B.

The Method

I employed the user port on the PET to transfer 8-bit data. Table 1 describes the user port signals. The CB2 and CA1 lines are used for handshaking the data. The sender sets the 8-bit port for output mode and the receiver for input mode.

Table 1: **User Port Signals**

PET Connections	Signal
A	Ground
B	CA1 - Input Handshake Line
C	Most Significant Data Line PA7
D	Data Line PA6
E	Data Line PA5
F	Data Line PA4
H	Data Line PA3
J	Data Line PA2
K	Data Line PA1
L	Least Significant Data Line PA0
M	CB2 - Output Handshake Line
N	Ground

The CB2 line from the sender is connected to the CA1 line of the receiver and acts as a Data Ready signal. The CB2 line from the reciever is connected to the CA1 line of the sender and acts as a Data Accepted signal. The wiring hookup is shown in figure 1.

I could have transferred the data from the BASIC program with PEEKs and POKEs. But for this application, I wrote a simple machine-lanugage program that transfers data much faster and allows the PETs to spend most of their time collecting data and doing numeric calculations, and very little time with the PET-to-PET communication.

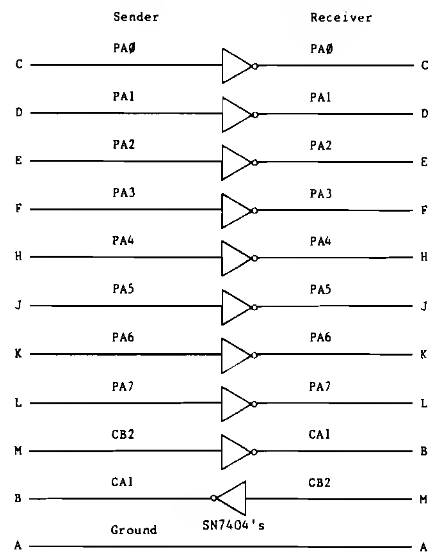
The data sent are the first 14 elements of the first dimensioned real array. This puts the restriction on the BASIC program that the first dimensioned array in the program is the one to be sent or received.

The set-up code for the sender (a SYS 637 command in the program listed) sets CB2 high, sets the data direction register for input, and clears CA1. When the sender wishes to send data, a SYS 634 is initiated in the sender code.

The set-up code for the receiver (a SYS 640 command in the program listed) sets CB2 high, sets the data direction register for input, and clears CA1. Also, the machine code changes the IRQ vector on the reciever to point to the machine-language routine that checks for a Data Ready signal from the sender.

The data are received in the receiver during the 60-Hz keyboard-scan routine, independent of action by the BASIC program. This is done by checking for a Data Ready from the sender each scan. If data are not ready, the normal keyboard scan functions as normal. If data are ready, the receiver code is executed, after which the keyboard-scan code continues. Because the data are received independently of the BASIC program, the receiver program must be able to determine whether or not new data have been sent. This is done by using the zero element of the array as a flag. The receiver sets the zero element to zero, and the sender

Figure 1. PET to PET Connection



sets it to minus one. These numbers are chosen because PET BASIC takes a value of zero in decisions to be false and a minus one to be true. In an IF statement the receiver PET can check the zero element. If it is minus one, new data have been sent and can be copied to a safe location and the zero element flag can be reset to zero.

Limitations

Although the sender PET can send information faster than the receiver PET needs it, in this application the sender spends most of its time collecting data and the receiver can plot them very quickly. This is not a problem if only the latest data are needed. If a future problem arises, additional coding in the program can be used to solve it. The additional machine code could check the zero element to see if it

is still minus one from the previous communication, in which case the receiver would not do the communication until it becomes zero.

The current program can be expanded only to send forty-nine elements of an array because the Y register of the 6502 microprocessor is used as a counter. This problem can be overcome by placing a two-byte counter in memory.

Description of Programs

The first three instructions in lines 1090 to 1110 of the machine code (listing 1) form a jump table. The next group of instructions in lines 1130 to 1170 set up the PET as a sender. After that, lines 1190 to 1250 set up the PET as a receiver. The PET IRQ routine for the receiver starts in line 1280. Lines 1270 to 1320 look for the first Data Ready from the sender by checking the CA1 interrupt flag. The macro in line 1350 loops for the number of bytes to receive. The receiver code waits for a Data Ready, gets the data, and sends a Data Accepted. Line 1420 is the macro that loops for the number of bytes to send. The sender code writes the data, sends a Data Ready, and waits for a Data Accepted. Lines 1500 to 1550 detects a Data Ready or Data Accepted. Data are read or written in lines 1730 to 1820, using the array pointer.

This machine code is for BASIC 2.0 and loads into Cassette Buffer 1. To use the code with BASIC 4.0, the keyboard scan address must be changed from \$E62E to \$E455 and the return to BASIC READY from \$C389 to \$B3FF.

The sample BASIC listing consists of two programs. Lines 100 to 260 form a sender program, and lines 270 to 380 form a receiver program. After the machine code has been loaded into both PETs, the BASIC program (listing 2) is run by the sequence given in the remarks in lines 120 to 160 of the program.

Conclusion

This program shows how easy it is to expand the firmware of the Commodore PET to implement new functions. EPROMs can be added to the hardware for these expanded firmware programs. This program also shows how machine language can improve the speed of the PET, and have a program function independently of a BASIC program.

Acknowledgements

The information contained in this article was developed during the course of work under Contract No. DE-AC09-76SR00001 with the U.S. Department of Energy.

You may contact Mr. Cochrane at E.I. du Pont de Nemours & Co., Savannah River Laboratory, Aiken, SC 29808.

Listing 1

```
0010 ;THE SENDER PET SENDS THE FIRST FIFTEEN (0-14) ELEMENTS OF
0020 ; THE FIRST DIMENSIONED ARRAY
0030
0040 ;THE RECEIVER PET RECEIVES DATA ALSO IN THE FIRST ARRAY
0050
0060 ; SYS 634 - SEND DATA
0070
0080 ; SYS 637 - SET UP SENDER PET
0090
0100 ; SYS 640 - SET UP RECEIVER PET
0110
0120 ;THE RECEIVER PET GETS DATA DURING THE KEYBOARD SCAN.
0130 ;THE RECEIVER PET CAN DETECT IF NEW DATA HAS BEEN RECEIVED
0140 ; BY CHECKING THE ZERO ELEMENT OF THE ARRAY. IF NEW DATA
0150 ; HAS BEEN RECEIVED THEN MOVE IT AND RESET THE
0160 ; ZERO ELEMENT.
0170
0180 ;IF THE PET HANGS UP AND THE STOP KEY DOES NOT FUNCTION
0190 ; USE THE  KEY TO RETURN TO READY.
0200
0210 POINTER    .DE 44              ;START OF ARRAYS
0220 IRQ        .DE $90             ;PET IRQ VECTOR
0230 READY      .DE $C389           ;WARM START OF BASIC
0240 PET.IRQ    .DE $E62E           ;PET IRQ ROUTINE
0250 PIAK       .DE 59410           ;KEYBOARD PORT
0260
0270 ;HEADER  BYTES/ELEMENT * ELEMENTS MOVED - ONE LESS
0280 ;  7          5        *      15       -    1
0290 COUNTER    .DE 775-1           ;BYTES TO TRANSFER
0300
0310 HDATA      .DE 59457           ;DATA WITH HANDSHAKE
0320 DDR        .DE 59459           ;DATA DIRECTION REG
0330 AUXREG     .DE 59467           ;AUXILLARY CONTROL REG
0340 PCREG      .DE 59468           ;PERIPHERAL CONTROL REG
0350 IFREG      .DE 59469           ;INTERRUPT FLAG REG
0360 DATA       .DE 59471           ;DATA REG
0370
0380 ;SET REGISTER 3 FOR INPUT OR OUTPUT
0390 !!!SET.DIR .MD (...DIR)
0400            LDA #...DIR
0410            STA DDR
0420            .ME
0430
0440 ;SET BITS IN SPECIFIED REGISTER WHICH CORRESPOND WITH
0450 ;1'S IN MASK.
0460 !!!SET.BIT .MD (...MASK ...REG)
0470            LDA #...MASK
0480            ORA ...REG
0490            STA ...REG
0500            .ME
0510
0520 ;CLEAR BITS IN SPECIFIED REGISTER WHICH CORRESPOND
0530 ;WITH 1'S IN MASK.
0540 !!!CLR.BIT .MD (...MASK ...REG)
0550            LDA #...MASK
0560            EOR #%11111111      ;INVERT MASK
```

Listing 1 *(Continued)*

```
0570            AND ...REG
0580            STA ...REG
0590            .ME
0600
0610 ;SET IRQ VECTOR TO NEW VALUE
0620 !!!SET.IRQ .MD (...VECTOR)
0630            SEI                 ;DISABLE IRQ'S
0640            LDA #L,...VECTOR
0650            STA *IRQ
0660            LDA #H,...VECTOR
0670            STA *IRQ1
0680            CLI                 ;RESTORE IRQ'S
0690            .ME
0700
0710 ;LOOP ROUTINE FOR SENDING AND RECEIVING
0720 !!!LOOP    .MD (...FIRST ...SECOND ...THIRD)
0730            SEI                 ;DISABLE IRQ'S
0740            LDY #COUNTER        ;SETUP POINTER TO AI
0750 ...AGAIN   JSR ...FIRST
0760            JSR ...SECOND
0770            JSR ...THIRD
0780            DEY                 ;DECREMENT THE COUNT
0790            BPL ...AGAIN        ;INCLUDES ZERO
0800            CLI                 ;RESTORE IRQ'S
0810            RTS
0820            .ME
0830
0840
0850            .BA 634             ;FIRST CASSETTE BUFI
0860            .CE                 ;CONTINUE IF ERRORS
0870
0880            .PR PET TO PET COMMUNICATION
0890
0900
0910            .PR STORE OBJECT CODE? (ONO, 1YES)
0920 OBJ        .IN OBJ
0930            IFN OBJ
0940            .OS
0950            ***
0960
0970            .PR LISTING OUTPUT? (ONO, 1YES)
0980 LISTIT     .IN LISTIT
0990            IFN LISTIT
1000            .LS
1010            .PR EXPAND MACROS? (ONO, 1YES)
1020 EXPAND     .IN EXPAND
1030            IFN EXPAND
1040            .ES
1050            ***
1060            ***
1070
1080 ;JUMP TABLE
027A- 4C EC 02  1090            JMP SEND.MAIN       ;SYS 634 FOR SENDINC
027D- 4C 83 02  1100            JMP BEGIN.S         ;SYS 637 FOR SENDER
0280- 4C 9E 02  1110            JMP BEGIN.R         ;SYS 640 FOR RECEIVI
                1120
```

Listing 1 *(Continued)*

```
                1130 BEGIN.S    SET.BIT (%11100000 PCREG) ;SET CB2 HIGH
0283- A9 E0
0285- 0D 4C E8
0288- 8D 4C E8
                1140            SET.DIR (%11111111)  ;SET FOR OUTPUT

0288- A9 FF
028D- 8D 43 E8
                1150            CLR.8IT (%00011100 AUXREG) ;DISABLE SHIFT RECIS

0290- A9 1C
0292- 49 FF
0294- 2D 48 E8
0297- 8D 48 E8

029A- AD 41 E8  1160            LDA HDATA          ;CLEARS CA1
029D- 60        1170            RTS
                1180
                1190 8EGIN.R    SET.8IT (%11100000 PCREG) ;SET CB2 HIGH

029E- A9 E0
02A0- 0D 4C E8
02A3- 8D 4C E8

                1200            SET.DIR (%00000000)  ;SET FOR INPUT

02A6- A9 00
02A8- 8D 43 E8

                1210            CLR.8IT (%00011100 AUXREG) ;DISABLE SHIFT REGIS

02AB- A9 1C
02AD- 49 FF
02AF- 2D 4B E8
0282- 8D 48 E8

02B5- AD 41 E8  1220            LDA HDATA          ;CLEARS CA1
                1230            SET.BIT (%00000001 AUXREG) ;ENABLE LATCHING OF

0288- A9 01
02BA- 0D 48 E8
02BD- 8D 48 E8

                1240            SET.IRQ (LOOK)     ;CHANGE PET IRQ VECTOR

02C0- 78
02C1- A9 CE
02C3- 85 90
02C5- A9 02
02C7- 85 91
02C9- 58

02CA- 60        1250            RTS
                1260
02C8- 4C 2E E6  1270 PETROUT    JMP PET.IRQ        ;TO PET IRQ ROUTINE
02CE- A9 02     1280 LOOK       LDA #%00000010
02D0- 2C 4D E8  1290            BIT IFREC
02D3- F0 F6     1300            8EQ PETROUT        ;CA1 NOT SET SO NO DATA RE
02D5- 20 DB 02  1310            JSR REC.MAIN
02D8- 4C C8 02  1320            JMP PETROUT        ;FINISH UP PET IRQ ROUTINE
                1330
                1340 ;MAIN RECEIVER ROUTINE
                1350 REC.MAIN   LOOP (WAIT.CA1 RECEIVE SEND.C82)

02DB- 78
02DC- A0 51
02DE- 20 FD 02
02E1- 20 27 03
02E4- 20 0A 03
02E7- 88
02E8- 10 F4
02EA- 58
02EB- 60

                1360
                1370 ;1. WAIT FOR DATA READY
                1380 ;2. GET DATA & CLEAR CA1
                1390 ;3. SEND DATA ACCEPTED
                1400
                1410 ;MAIN SENDER ROUTINE
                1420 SEND.MAIN  LOOP (SENDER SEND.CB2 WAIT.CA1)

02EC- 78
02ED- A0 51
02EF- 20 1F 03
02F2- 20 0A 03
02F5- 20 FD 02
02F8- 88
02F9- 10 F4
```

Listing 1 *(Continued)*

```
02FB- 58
02FC- 60
                1440 ;1. WRITE DATA & CLEAR CA1
                1450 ;2. SEND DATA READY
                1460 ;3. WAIT FOR DATA ACCEPTED
                1470
                1480
                1490 ;WAIT FOR CA1 TO 8E SET
02FD- A9 02     1500 WAIT.CA1   LDA #%00000010     ;MASK TO READ CA1
02FF- 2C 12 E8  1510 LOOP       BIT PIAK           ;TEST FOR PANIC KEY
0302- 10 29     1520            BPL ESCAPE         ;IF  KEY THEN SIGN BIT S
0304- 2C 4D E8  1530            BIT IFREC
0307- F0 F6     1540            BEQ LOOP           ;LOOP IF CA1 NOT SET
0309- 60        1550            RTS
                1560
                1570 ;SEND DATA READY OR DATA ACCEPTED
030A- 20 16 03  1580 SEND.C82   JSR TOGGLE.CB2
030D- EA        1590            NOP                ;DELAY SOME
030E- EA        1600            NOP
030F- EA        1610            NOP
0310- EA        1620            NOP
0311- EA        1630            NOP
0312- 20 16 03  1640            JSR TOGGLE.CB2
0315- 60        1650            RTS
                1660
                1670 ;SET CB2 TO REVERSE STATE
0316- A9 20     1680 TOGGLE.C82 LDA #%00100000     ;MASK FOR C82 OUTPUT CON
0318- 4D 4C E8  1690            EOR PCREG          ;TOGGLE 8IT 5
031B- 8D 4C E8  1700            STA PCREG          ;WHICH CHANGES CB2 HIGHL
031E- 60        1710            RTS
                1720
                1730 ;SET DATA TO SEND OUT
031F- 81 2C     1740 SENDER     LDA (POINTER),Y
0321- 49 FF     1750            EOR #%11111111     ;INVERT FOR INVERTERS
0323- 8D 41 E8  1760            STA HDATA          ;CLEARS CA1
0326- 60        1770            RTS
                1780
                1790 ;STORE DATA RECEIVED
0327- AD 41 E8  1800 RECEIVE    LDA HDATA          ;CLEARS CA1
032A- 91 2C     1810            STA (POINTER),Y
032C- 60        1820            RTS
                1830
                1840
                1850 ;ESCAPE CODE IF PANIC KEY () PRESSED
032D- 58        1860 ESCAPE     CLI
032E- 4C 89 C3  1870            JMP READY          ;RETURN TO BASIC
                1880
                1890
                1900            .EN
END OF MAE PASS!

 --- LABEL FILE: ---

AUXREC E848          8EGIN.R 029E         8EGIN.S 0283
COUNTER 0051         DATA E84F            DDR E843
ESCAPE 032D          EXPAND 0001          HDATA E841
IFREC E84D           IRQ 0090             LISTIT 0001
LOOK 02CE            LOOP 02FF            08J 0000
PCREG E84C           PET.IRQ E62E         PETROUT 02CB
PIAK E812            POINTER 002C         READY C389
REC.MAIN 02D8        RECEIVE 0327         SEND.C82 030A
SEND.MAIN 02EC       SENDER 031F          TOGGLE.CB2 0316
WAIT.CA1 02FD
//0000,0331,0331
```

Listing 2

```
 100 REM PET TO PET TEST PROGRAM
 110 REM SENDER PROGRAM
 120 SYS 637:REM SET UP FOR SEND
 130 PRINT"■■■■CONT■■■■":STOP
 140 DIM X(6):REM DEFINE ARRAY TO SEND
 150 FOR I=0 TO 6:READ X(I):NEXT:REM LOAD ARRAY
TO SEND
 160 SYS 634:REM SEND DATA
 170 END
 180 DATA -1,1,2,3,4,5,6
 190 REM
 200 REM RECEIVER PROGRAM
 210 SYS640:REM SET UP FOR RECEIVING
 220 PRINT"■■■■CONT■■■■":STOP
 230 DIM X(6):X(0)=0:TB=34
 240 FORI=1 TO 6:X(I)=0:NEXT
 250 PRINT"■■■CURRENT ARRAY ELEMENTS■"
 260 FORI=1 TO 6:PRINTX(I):NEXT
 270 IF X(0) THEN PRINT"■":FOR I=1 TO 6:PRINTX(I)
:NEXT:PRINT"■■ARRAY RECEIVED":X(0)=0
 280 PRINT"■";TAB(TB);"■";TI$;
 290 GOTO270
```

MICRO

Multi-Microprocessor Tidbits

by Mike Rosing

Running a 6502 and 6809 in the same computer simultaneously creates a powerful device. This article describes problems you might encounter and a general description of a specific task for which two processors were used.

Watching two 300-baud lines simultaneously and recalling each record that comes over those lines is easy with a multiprocessing system. By using two Asynchronous Communication Interface Adaptors (ACIAs) connected to an Apple's Interrupt Request Line (IRQ) and a Stellation Two 6809 board, the data collection is done in background and the data display is done in foreground.

Some problems running two micros simultaneously include waking up, communication, and debugging. The major problem is finding a 6809 assembler for the Apple. At the time I purchased the Stellation Two board there was no software. Now you can get a very nice assembler and debugger from Stellation Two for about $150.

I bought the assembler package that runs under the UCSD p-System from Softech Microsystems. It works on the Apple Pascal system but is difficult to transfer from the 8-inch floppy (with no paper work to tell how to read the disk) to the Apple 5¼-inch floppy. The assembler also has several bugs. For $12,000 Softech will release the source listing but they won't fix the bugs for you!

The hardware consists of an Apple II with a 16K board in slot zero. The board was modified by breaking a tie and soldering a circle on the Apple 16K board to allow use of 2716 EPROMs. When the Apple is turned on the 2716 holds the reset vector enabling the Apple to become a dedicated machine.

The Stellation Two 6809 board has an EPROM slot built in so no modification of that board is necessary. The ACIAs are mounted on an Apple prototype board along with a few chips for buffers and logic for chip selection.

Each 300-baud line is terminated in a line receiver chip. The receiver outputs go to two Synertek ACIAs. After building the board with two crystals I learned that four ACIAs could be run with one crystal by using the clock outputs on the chips and programming the ACIAs correctly. It is possible to talk and listen to four serial lines using the multiprocessing system described here.

The wake-up routine for each computer is different. When Reset is pressed the 6502 is on and the 6809 is off. The 6502 executes the following code to turn the 6809 on (note that all interrupt lines are high before the 6809 is turned on):

```
SLOT    EQU 70           ;6809 slot pos. (ex.)
IRO02   EOU C080 + SLOT  ;6502 IRQ line
HALT    EQU C081 + SLOT  ;6809 halt line
RESET   EQU C082 + SLOT  ;6809 reset line
NMI     EOU C083 + SLOT  ;6809 non-maskable
                          interrupt line
FIRQ    EQU C084 + SLOT  ;6809 fast interrupt ln
IRQ     EQU C085 + SLOT  ;6809 interrupt reqst ln
ROM     EOU C086 + SLOT  ;on bd ROM enable bit
                          for Stellation Two
SWAP    EQU C087 + SLOT  ;switches A15 to be
                          opposite or same
                          as 6502
;most significant bit of each location determines
 what lines will do

LDA #0       ;ensure that
STA HALT     ;6809 is
STA RESET    ;off
STA IRO02    ;6502 interrupt goes out invert gate
LDA #80      ;raise
STA FIRQ     ;all 6809
STA IRO      ;interrupt
STA NMI      ;lines
STA SWAP     ;tells 6809 bd that A15 isn't flipped
             ;both CPU s view RAM the same way
STA ROM      ;80 - ROM slot used, 00 - not used
STA HALT     ;6809 on and
STA RESET    ;going through reset procedure
```

When the 6502 reaches the last instruction the 6809 is on and running. The 6502 goes at about 1/5th its normal pace and the 6809 goes at full speed.

The 6809 wake-up routine is simple. As shown below, the 6809 defines its stacks, turns on the ACIAs and then unmasks its IRQ line.

```
WAKEUP ORCC #5OH       ;mask interrupts
       LDU #USRSTK     ;set up
       LDS #SYSSTK     ;stack pointers
       CLR STATUS      ;set up
       CLR STATUS + 4  ;ACIAs with
       LDA #16H        ;1 stop, 8 data bits
       STA CNTRL       ;300 baud
       STA CNTRL + 4   ;no parity
       LDA #1          ;receiver interrupt
       STA CMD         ;enabled
       STA CMD + 4     ;transmitter disabled
       CLRA            ;set direct page
       TFR A,DP        ;same as Apple's zero pg
       ANDCC #OEFH     ;enable IRQ
```

The addresses used depend on the logic used to get to each ACIA. These can be set using equates at the beginning of the code file.

The background task of collecting data from two serial lines is accomplished using interrupts from the ACIAs to the 6502 and the IRQ line from the 6502 to the 6809. This allows the operator to view call data from two hours ago at the same time new calls are coming in.

Once eight bits have been collected, either ACIA pulls the IRQ low to the 6502. The 6502 vectors to the interrupt handler and checks each ACIA to see which one is requesting service. If both ACIAs are requesting service, then IRQ will not clear and the 6502 will vector to the interrupt handler again. At 300 baud there is no loss of data for an interrupt handler that takes less than 30

milliseconds. When the 6809 is the master computer, the 6502 runs at about 1/5th normal speed. An average instruction takes four clock cycles on the 6502. Taking 5*1E−6 seconds as a clock cycle and 4*5E−6 seconds as an instruction (on average), the total number of instructions before loss of data is 3E−2/2E−5 = 1.5E+3. The interrupt handler in my system uses only 50 instructions. This allows plenty of time for foreground.

The beginning of the interrupt handler for the 6502 is shown below. After saving the registers, each ACIA must be polled to find which one is requesting service. Reading the status register of the 6551 ACIA clears the interrupt. The most significant bit tells the 6502 if the interrupt came from the device polled.

```
INTRPT  PHA           ;save
        TYA           ;all
        PHA           ;registers
        TXA
        PHA
        BIT P1STUS    ;port 1 status checked
        BMI BOX1      ;if N bit set then ACIA 1
                       gave interrupt
        BIT P1STUS+4  ;port 2 status checked
        BMI BOX2      ;if N bit set then ACIA 2
                       gave interrupt
        LDA ERMSG     ;if neither set then there
                       was an error
        JMP PRNTMSG   ;so tell operator and
                       then stop
```

After saving the byte into the buffer and incrementing the buffer pointer, the 6502 pulls all registers from the stack and executes RTI. The error message at the end is for debugging purposes. The IRQ from the 6809 to the 6502 goes through an inverting gate; this caused some problems before discovery.

At the end of each serial line the computer sends a start of text (STX) and end of text (ETX) for each message. The 6502 reads an entire message from STX to ETX and saves this to an input buffer. Upon receiving an ETX, it saves the line number in a common location and pulls the 6809 IRQ. The 6809 checks which line has sent a completed message and then processes that buffer. There are many choices for the 6809, so its interrupt handler is over 250 instructions. Since the 6809 is the master CPU it takes about the same time as 50 instructions on the 6502. The 6809 also has more foreground tasks to do than the 6502. Both programs fit in 2K EPROMs. The rest of memory is used for record storage.

The beginning of the 6809 interrupt handler is shown below. A mailbox system is used to tell the 6809 which buffer to take care of. Since the IRQ is masked on vectoring to the interrupt, levels of interrupt are not allowed without unmasking. Because the operation of two 300-baud lines is slow, no attempt was made to make this system that complex.

```
INTRPT LDA #80H   ;raise
       STA IRQ    ;6809 IRQ line
       LDA IJOB   ;box 1 or 2?
       STA TJOB   ;save in case another
                   interrupt is coming in fast
       BNE BOX2   ;non zero for box 2, zero
                   for box 1
       LDD B1IB   ;X reg is
       BRA GTBUF  ;input buffer
BOX2   LDD B2IB   ;pointer
GTBUF  EXG A,B    ;swap byte sex (this is
                   important!)
       TFR D,X    ;now have input buffer ptr
       NEGB       ;go back to first char
       LDA B,X    ;get first char in buffer
```

The location IJOB is the mailbox. TJOB is a temporary storage location in case another interrupt is attempted from the other box. B1IB (Box 1 Input Buffer) and B2IB are 6502 zero-page pointers that tell the 6502 where to put the next input character. The 6809 uses these as pointers to the input buffers as well as for the length of the message in the buffer. By incrementing the B register and leaving the X register alone, it is easy to step through the input buffer. When B is zero, the end of the buffer has been reached.

The memory is organized with two 256-byte buffers for the input messages. Above those are two 1K buffers for "live calls." These are 32 slots (one for each phone line), which are 32 bytes each. When a call is finished, the slot corresponding to that line is packed in BCD format into the top of memory. This region is actually a ring buffer that holds about 2400 calls. As more calls come in, old calls are lost.

The operator can examine either live calls or past calls by using menu commands. The 6502 constantly polls the keyboard in foreground and when a key is pressed the processor compares the key to the acceptable commands. The 6502 then jumps to the routine that gathers the data the 6809 foreground program needs. For example, searching past records for all calls to area code 307 requires the 6502 to put the message "AREACODE?" on the screen. The 6502 then reads the keyboard for the area code and saves it to a common zero-page location. The 6809 is constantly checking a common location known as a mailbox. As long as the mailbox is zero the 6809 foreground has little to do. Once the 6502 gets the area code into a common buffer it puts a job number into the mailbox. The 6502 then goes to an input routine that controls the paging of records (since only 24 lines are visible on the screen at a time).

"When the 6809 is the master computer, the 6502 runs at about 1/5th normal speed."

The foreground codes for the 6502 and 6809 are similar. The 6502 scans the keyboard location to see if any key has been pressed. The 6809 scans a mailbox to see if any jobs have been requested. In the meantime the background is running *via* interrupts. The

6809 foreground code is shown in the next code listing. Once the 6502 has collected all the data from the operator, it sends the 6809 a job number *via* a JOBBOX. The 6809 continuously scans the JOBBOX until a non-zero value appears *via* the 6502. It uses this job number as an index into a pointer table, and the job is executed as a subroutine. The code is written as relocatable, which really is not necessary for the job at hand. (This is only one of many ways to communicate between the two computers.)

```
JOBBOX  EQU 4            ;zero pg for both
                          6502 and 6809
START   LDA JOBBOX       ;any jobs?
        BEQ START        ;loop till there is
        ASLA             ;convert job number
                          to offset
        LEAX JMPVEC,PCR  ;get tbl address into
                          X reg
        LDX A,X          ; get relative offset
                          into code
        LEAY WAKEUP,PCR  ;get actual start of
                          code location
        TFR Y,D          ;relative offset plus
        JSR D,X          ;starting loc gives
                          absolute position
        BRA START        ;look for next job
JMPVEC  WORD DUMMY       ;never used
        WORD JOB1        ;when assembled will
        WORD JOB2        ;hold offsets into
        WORD JOB3        ;file from zero pos
        .....            ;which was wakeup
                          in this case
```

Once the 6809 gets a job number it jumps to the routine requested. In this case it packs the area code sent by the 6502 into BCD format and then scans all of the ring buffer for calls matching that area code. On a match the record is unpacked onto the screen. When 24 records have been found the 6809 waits for the 6502 to send a go signal to keep looking. Once all of the ring buffer has been scanned, both 6502 and 6809 return to polling their respective memory locations for the next foreground job. Meanwhile the background is still recording information coming over the two lines.

The 6809 can scan for input lines, output lines, authorization codes, and status messages, as well as area code. Each of these are part of a call record. The routines that scan memory use some common subroutines for bumping from one record to the next in the ring buffer. The Stellation Two board can support a 4K EPROM, but only 2K is needed for this dedicated application.

Choosing what each processor should do is arbitrary. The system described here uses the 6502 for interactive I/O operations and the 6809 for all memory tasks. I find the 6809 easier to program than the 6502. Whether or not one microprocessor could do all the above as fast as two is not clear

The 6502 routine uses the 16K of RAM on the card as well as the 2K EPROM. By writing itself onto the RAM and then throwing the soft switch that allows the RAM to be read/write, the full 16K is available. The 6809 uses the bottom part of this RAM for its stack, leaving the 48K of RAM on the mother board for buffers. The code that does this follows:

```
RAMWRT  EQU C089         ;write enable RAM cd
RAMRD   EQU C08B         ;read enable RAM cd

TRONST  LDA #0FF         ;will be zero
        BEQ TRON2X       ;on warm start
        LDA RAMWRT       ;write enable
        LDA RAMWRT       ;RAM cd while getting
                          code from EPROM
        LDA #0           ;clear index counter
        TAY
        STA 0            ;set zero pg ptr to start
                          of ROM
        LDA #0F8         ;which is F800
        STA 1            ;up to FFFF
$1      LDA @0,Y         ;get a byte from ROM
        STA @0,Y         ;copy into RAM!
        INY              ;bump counter
        BNE $1           ;bump
        INC 1            ;zero-page counter
        BNE $1           ;until past FFFF
        LDA #0           ;set RAM for warm
                          reset
        STA TRONST+1     ;because we don't
                          need to do this again
TRON2X  LDA RAMRD        ;read/write
        LDA RAMRD        ;enable RAM
        ; at this point the EPROM is not used
        ; but its code is running in RAM on the
        ; 16K board. On a warm reset the
        ; above code is bypassed since the
        ; reset vector is unchanged but the
        ; branch instruction will see zero
```

Debugging the above system required putting out messages on the screen to state how far into its program each computer had gotten. When I put 6502 messages at the top of the screen and 6809 messages at the bottom, the problem point was found easily. Usually the problems I had were byte-sex related or mailboxes not at the same address. By clearly separating the tasks of the two processors, mistakes and bugs can be found relatively quickly.

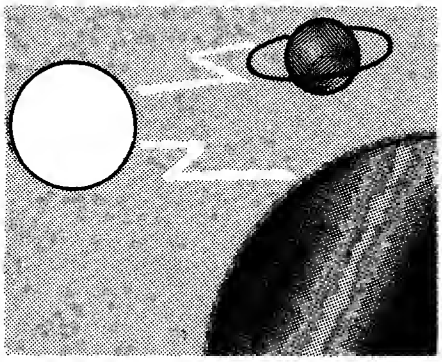
The specific examples used above work. They are not necessarily the only way to do multiprocessing in a dedicated environment. If you spend time deciding what each computer should do, the power of multiprocessing will become apparent.

You may contact Mike Rosing at 4260 E. Evans Ave., Denver, CO 80222.

MICRO™

In-House Communication

by Phil Daley



MICRO has always been in the forefront of disseminating information useful to computerists of many types of systems. This necessitates our having different kinds of hardware, disks, and tape formats. In addition, the staff must know many different languages and dialects. To help overcome this drawback, we have established a centralized system that other computers can "talk" to and, eventually, receive from.

At MICRO, we have set up a system that utilizes a 6809 computer, the FOCUS, as the end source of all our files, and a word-processing system called TYPE+, written by Bob Tripp. An interface to a Compugraphic Editwriter allows us to transfer text files to the phototypesetter without human intervention, and to typeset those files without further editing. This is made possible by preprocessing the text files with the TYPE+ program.

Author-submitted and in-house articles and programs are written on whatever computer is available and appropriate to the task at hand. Then they are sent to the FOCUS using the Stylograph text editor, entry mode. This program takes the text or listing in from the serial port and stores it line by line in the 6809 RAM (see listing 1), then the file is saved to disk. The Stylograph text editor imposes two restrictions on the listings: the first character on a line cannot be a "#", and the input buffer on the Flex operating system must not exceed 128 characters, including the carriage return. A line that is too long results in a carriage return being unaccepted, and the remainder of the file continuing to overflow the buffer. Although you will have quite a mess on the screen, you need only delete the current line to enable a normal SAVE operation.

The TYPE+ program includes a word processor that has the several Editwriter keyboards encoded to special keys on the FOCUS, enabling screen display of all the special Editwriter functions. In addition to the preprocessing function, we use the FOCUS as an additional Compugraphic terminal for normal typesetting input.

Since the Editwriter uses different ASCII codes for display than a standard computer, and has several dozen extra keys and codes, it is necessary to convert many of the standard codes in the text file to the non-standard Editwriter format. In addition, the display uses standard ASCII whenever possible, so the normal keys have to be converted from standard display to Editwriter display when transferring the file.

The standard file includes special Editwriter information so that the Compugraphic will understand what to do with the file when it arrives. Such things as font number and type size have to be specified at the beginning of the file and whenever any of the parameters have to be changed. A SEARCH and REPLACE function substitutes the required Editwriter codes for each regular character that has to be changed. For instance, the Editwriter will not accept the double quote ("). Each occurrence of the " is replaced by lower precedent 0 ($\frac{P}{R}\downarrow$ 0). This ensures that the quote will appear as the proper code when the transfer takes place.

The most complicated change involves the 0. The SUN-MOON listing in January (MICRO 56:36) used the variable O extensively throughout the listing. When I proofread the listing, I couldn't see any difference in the O and the 0 (although the production people could). I thought that anyone trying to key in the program would be unable to notice the distinction.

I learned that the Compugraphic has a command called "Flash Only," which means that the character is printed but the paper is not advanced. This allows over-striking: the / is printed without advancing and then the 0 is printed on top of it. Simple in theory, unfortunately it turns out that this causes the slash to appear too low in the 0 to look natural. However, another command on the Compugraphic allows a character to be raised or lowered any number of points (plus or minus a point of lead $\frac{+P}{LD}$ $\frac{-P}{LD}$). With this command, you can raise the slash in the zero to the center. The final substitution becomes: replace 0: with minus a point of lead, flash only, /, plus a point of lead, 0.

After adjusting the non-allowable characters to Compugraphic character codes, and the line lengths to the proper size for publication, a short program called TRANSFER is invoked to LIST the program to the Editwriter where it is entered as a file. The interface to the FOCUS has the Editwriter thinking that someone is typing the file into the keyboard instead of being sent through the serial port. The received file is then rejustified and saved to disk to be output in the normal manner when needed.

We are currently working on a program that will take a previously defined glossary and make all the necessary changes to the text file automatically. This will increase our productivity and, at the same time, decrease our typographical errors (when the bugs are out).

The Bulletin Board

The MICRO Bulletin Board System is working well and we have many regular callers. The BBS runs on our Apple II, but may be called by anyone with a modem. It normally runs four days a week, Monday through Thursday from 5:00 p.m. to 8:00 a.m. We are moving our offices and do not have the new phone number yet, but will let you know in our May issue. Anyone may call the system, but only subscribers are issued passwords for writing on the system. There are several useful programs that users may download onto their own systems, and we hope to have a selection for different machines before too long. If anyone has a program (personal or public domain) they would like to see get wider distribution, send it to us (*via* the BBS) and we will put it on-line.

Articles also can be received through this system and we have on-line capabilities with COMPUSERVE and THE SOURCE. An author can download to them and we can retrieve the file. (We received part of Clifford Glennon's communication article this way.) There are a few bugs to be worked out to make this a viable alternative; the lower-to-upper-case conversion and maximum file-length restriction are two.

You may contact Phil at MICRO, P.O. Box 6502, Chelmsford, MA 01824.

```
KEYBOARD/ACIA INPUT DRIVER          1-14-83 TSC ASSEMBLER

                        * TERMINAL DRIVER PROGRAM FOR FLEXI-PLUS

                        *  27 MARCH 1982 - FOCUS VERSION - JR
                        *  27 MARCH 1982 - BAUD RATE SELECTION ADDED - RMT
                        *  14 APRIL 1982 - MODIFIED FOR APPLE - RMT
                        *   7 MAY   1982 - BASED ON APPLE.TXT - RMT+JJR
                        *   3 NOVEM 1982 - MODIFIED TO CMD - PRD

                        * EQUATES

                   CD03 WARMS   EQU   $CD03
                   CD42 GETHEX  EQU   $CD42

                   F829 VKIN    EQU   $F829
                   F82C VKSTAT  EQU   $F82C
                   E549 STDIN   EQU   $E549

                   F815 MRSTAT  EQU   $F815

                   E180 ADATA   EQU   $E180
                   E181 ASTAT   EQU   ADATA+$01
                   E182 ACMD    EQU   ASTAT+$01
                   E183 ACTRL   EQU   ACMD+$01

 C100                           ORG   $C100

 C100 BD   CD42         APPLE2  JSR   GETHEX    GET USER'S BAUD RATE
 C103 CC   0B18                 LDD   #$0B18    1200 BAUD
 C106 8C   1200                 CMPX  #$1200    IF NOT 1200
 C109 27   02                   BEQ   NOADJ     THEN DEFAULT TO
 C10B C6   16                   LDB   #$16      300 BAUD
 C10D 8D   1F           NOADJ   BSR   VAINIT
 C10F 30   8D 000D              LEAX  INPUT,PCR
 C113 BF   E54A                 STX   STDIN+1
 C116 30   8D 002D              LEAX  STATUS,PCR
 C11A BF   F816                 STX   MRSTAT+1
 C11D 7E   CD03                 JMP   WARMS     GO BACK TO FLEX

 C120 BD   F829         INPUT   JSR   VKIN      NO
 C123 25   01                   BCS   RTEST
 C125 39                        RTS

 C126 8D   0D           RTEST   BSR   VRCVR
 C128 25   03                   BCS   KTEST
 C12A 4D                        TSTA
 C12B 27   F3                   BEQ   INPUT     IGNORE NULLS
 C12D 39                KTEST   RTS
 C12E FD   E182         VAINIT  STD   ACMD
 C131 B6   E180                 LDA   ADATA     READ OLD DATA
 C134 39                        RTS

 C135 34   04           VRCVR   PSHS  B
 C137 53                        COMB            SET CARRY TO INDICATE
                                                NO DATA (YET)
 C138 F6   E181                 LDB   ASTAT
 C13B C4   08                   ANDB  #$08
 C13D 27   06                   BEQ   NODATR
 C13F B6   E180                 LDA   ADATA
 C142 84   7F                   ANDA  #$7F      STRIP PARITY
 C144 5F                        CLRB            CLEAR CARRY INDICATES DATA RCVD
 C145 35   84           NODATR  PULS  B,PC

 C147 BD   F82C         STATUS  JSR   VKSTAT
 C14A 24   03                   BCC   RETURN
 C14C BD   F815                 JSR   MRSTAT
 C14F 39                RETURN  RTS

                                END   APPLE2
```

BULLETIN BOARD

ABBS ABACUS II, Toledo, OH	(419) 865 1594	
ABBS AGS, Atlanta GA	(404) 733 3461	*24
ABBS Akron Digital Group, Akron, H	(216) 745 7855	*24
ABBS Apple Bin Washington	(206) 937 0444	
ABBS Apple Crate I, Seattle, WA	(206) 935 9119	
ABBS Apple Crate II, Seattle, WA	(206) 244 5438	
ABBS Apple-Med, Iowa City, IA	(319) 353 6528	
ABBS Apple-Mate, New York, NY	(201) 864 5345	
ABBS Baileys Computer Store, Augusta, GA	(404) 790 8614	
ABBS Baton Rouge, LA	(504) 291 1360	
ABBS Byte Shop, Ft. Lauderdale, FL	(305) 486 2983	
ABBS Byte Shop, Miami, FL	(305) 261 3639	
ABBS Calvary Mission Church, Mnpls, MN	(612) 471 0252	-rl
ABBS CCNJ, Pompton Plains, NJ	(201) 835 7228	
ABBS Century Next Computers, St. Louis, MO	(314) 442 6502	
ABBS Charlotte, NC	(704) 364 5254	
ABBS CODE, Glen Ellyn IL	(312) 537 7063	*24
ABBS Colortron Computer, WI	(414) 637 9990	*24
ABBS Compumart, Ottawa, Ontario, Canada	(613) 725 2243	
ABBS Computerland, Fremont, CA	(415) 794 9314	
ABBS Computer Corner, Amarillo, TX	(806) 355 5610	
ABBS Computer Conspiracy, Santa Monica, CA	(213) 829 1140	
ABBS Computer Crossroads, Columbia, MD	(301) 730 0922	
ABBS Computer Lab, Memphis, TN	(901) 761 4743	
ABBS Computer Room, Kalamazoo, MI	(616) 382 0101	
ABBS Computer Store, Toledo, OH	(419) 531 3845	
ABBS Dallas Info Board	(214) 248 4539	
ABBS Denver, CO	(303) 759 2625	
ABBS Detroit, MI	(313) 477 4471	
ABBS Electro-Mart, Spokane, WA	(509) 534 2419	*24
ABBS Fort Walton Beach, Destin, FL	(904) 243 1257	
ABBS Gamemaster, Chicago, IL	(312) 475 4884	*24
ABBS Hayward, CA	(415) 881 5662	
ABBS Illini Microcomputer Naperville, IL	(312) 420 7995	
ABBS Jonathan's Marlton, NJ	(609) 983 5970	
ABBS Ketchikan, AK	(907) 225 6789	
ABBS Livingston, NJ	(201) 994 9620	*24
ABBS Long Island, NY	(212) 448 6576	
ABBS Louisville, KY	(502) 426 2975	
ABBS Madam Bokeatha Society, Houston, TX	(713) 455 9502	
ABBS Michigan Apple-Fone, Southfield, MI	(313) 357 1422	
ABBS Newport Beach, CA	(714) 645 5256	*24
ABBS Omaha, NE	(402) 339 7809	
ABBS PCnet, San Francisco, CA	(415) 863 4703	*24
ABBS Pacific Palasades, Los Angeles, CA	(213) 459 6400	
ABBS Peoria, IL	(309) 692 6502	
ABBS Philadelphia, PA	(215) 628 3134	
ABBS Phoenix, AZ	(602) 898 0891	
ABBS Pirates Cove, Long Island, NY	(516) 698 4008	
ABBS Rogers Park, Chicago, IL	(312) 973 2227	
ABBS Software Sorcery, Herndon, VA	(703) 471 0610	
ABBS South of Market, San Francisco, CA	(415) 469 8111	-so
ABBS St. Louis, MO	(314) 838 7784	*24
ABBS Teledunjon I, Dallas, TX	(817) 469 1626	
ABBS Teledunjon II, Dallas, TX	(214) 530 0858	
ABBS Teledunjon III, Dallas, TX	(214) 960 7654	
ABBS The Moon, Dallas, TX	(214) 931 3437	*24
ABBS Turnersville, NJ	(609) 228 1149	
ABBS Vancouver, B.C.	(604) 437 7001	
ABBS Vermont, Essex Junction, VT	(802) 879 4981	*24
ABBS West Palm Beach, FL	(305) 848 3802	
ABBS Rob Roy Computer, Yakima, WA	(509) 575 7704	
ABBS Youngs Elect Svc., College Station, TX	(713) 693 3462	*24
ABBS #X, Atlanta, GA	(404) 256 1549	

A-C-C-E-S-S Annapolis, MD	(301) 267 7666	*24
A-C-C-E-S-S Olympia, WA	(206) 866 9043	*24
A-C-C-E-S-S Phoenix, AZ	#1 (602) 996 9709	*24
A-C-C-E-S-S Phoenix, AZ	&(602) 957 4428	*24
A-C-C-E-S-S Phoenix, AZ	(602) 274 5964	
A-C-C E S-S Scotsdale, AZ	(602) 998 9411	*24
A-C-C-E-S-S Wyckoff, NJ	(201) 891 7441	*24

AMIS A.R.C.A.D.E. Sterling Heights, MI	(313) 978 8087	*24
AMIS Chicago, IL	(312) 789 3610	*24
AMIS APOGEE Miami, FL	(305) 238-1231	-rb
AMIS GRAFEX Cupertino, CA	(408) 253 5216	
AMIS G.R.A.S.S. Grand Rapids, MI	(616) 241 1971	*24
AMIS IBBBS San Jose, CA	(408) 298 6930	
AMIS M.A.C.E. Detroit, MI	#1 (313) 868 2064	*24
AMIS Magic Lantern, Madison, WI	(608) 251 8538	
AMIS SB-12 Boston, MA	(617) 876 4885	
AMIS Space Seattle, WA	(206) 226 1117	
AMIS Starbase 12 Philadelphia, PA	(215) 876 8854	
AMIS T.A.B.B.S. Sunnyvale, CA	(408) 942 6975	

ARMU FLEGLG New York, NY	(212) 598 0719	
ARMU GREKLCOM Oklahoma City, OK	(405) 722 5056	
ARMU PACE Pittsburg, PA	(412) 655 3046	

ARM UDIC Washington, DC	#1 (202) 276 8342	
ARM UDIC Computer Age, Baltimore, MD	(301) 587 2132	

BBS IBM Hostcomm Atlanta, GA	!(404) 252 4146	
BBS IBM Hostcomm Claremont, CA	!(714) 624 1767	
BBS IBM Hostcomm Fairfax, VA	!(703) 978 9592	*24
BBS IBM Hostcomm Fairfax, VA	!(703) 978 0921	*24
BBS IBM Hostcomm Fairfax, VA	!(703) 591 5120	*24
BBS IBM Hostcomm Fairfax, VA	!(703) 425 9452	*24
BBS IBM Hostcomm Springfield, VA	!(703) 425 7229	*24
BBS IBM Hostcomm Houston, TX	!(713) 890 0310	
BBS IBM Hostcomm Toronto, Ontario, CN	!(416) 499 7023	*24

BBS IBM PC Atlanta, GA	!(404) 294 6879	
BBS IBM PC Atlanta, GA	!(404) 252 9438	*24
BBS IBM PC Beltsville, MD	!(301) 937 4339	*24
BBS IBM PC Bethesda, MD	!(301) 460 0538	*24
BBS IBM PC Billings, MT	!(406) 656 9624	
BBS IBM PC California Users Group	!(805) 987 4127	*24
BBS IBM PC Dale City, VA	!(703) 680 5220	*24
BBS IBM PC Denver, CO	!(303) 773 2699	*24
BBS IBM PC Gaithersburg, MD	!(301) 251 6293	*24
BBS IBM PC Madison, WI	!(608) 262 4939	*24
BBS IBM PC New York, NY	!(201) 678 6670	*24
BBS IBM PC Rockville, MD	!(301) 949 8848	*24
BBS IBM PC Vienna, VA	!(703) 560 7803	*24

BBS IBM PCmodem Chicago, IL	!(312) 259 8086	*24

BULLET-80 Boston, MA	&(617) 266 7789	*24
BULLET-80 Chesterland, OH	(216) 729 2769	
BULLET-80 Clarks Summit, PA	(717) 586 2112	
BULLET-80 Danbury, CT	#1(203) 744 4644	
BULLET-80 Fayetteville, GA	(404) 461 9686	
BULLET-80 Hawkins, TX	(214) 769 3036	
BULLET-80 Holstein, IA	(712) 368 2651	
BULLET-80 Houston, TX	(713) 331 2599	
BULLET-80 Ironton, OH	(614) 532 6920	
BULLET-80 Laguna Hills, CA	(714) 770 5052	
BULLET-80 Langhorne, PA	(215) 364 2180	
BULLET-80 Littlefield, TX	(806) 385 6843	
BULLET-80 New York, NY	(212) 740 5680	*24
BULLET-80 Orange County, Anaheim, CA	(714) 952 2110	
BULLET-80 San Jose, CA	(408) 241 0769	
BULLET-80 Seymour, CT	(203) 888 7952	
BULLET-80 Springfield, IL	(217) 529 1113	
BULLET-80 Tulsa, OK	(918) 749 0059	*24
BULLET-80 Waterford, MI	(313) 683 5076	*24

CBBS AMRAD, Washington, DC	(703) 734 1387	*24
CBBS Atlanta, GA	(404) 394 4220	*24
CBBS Baton Rouge, LA	(504) 273 3116	*24
CBBS Bloomington, IN	(812) 334 2522	
CBBS Boston, MA	(617) 646 3610	*24
CBBS Cedar Rapids, IA	(319) 364 0811	
CBBS Chicago, IL	#1 (312) 545 8086	*24
CBBS Corpus Christi, TX	(512) 855 1512	
CBBS CPEUG/ICST Gaithersburg, MD	(301) 948 5717	
CBBS CP/M Long Island, NY	(516) 698 8619	-rb
CBBS Lambda, Berkeley, CA	(415) 658 2919	-so
CBBS Lawrence General Hospital, Boston, MA	(617) 683 2119	
CBBS LICA LIMBS, Long Island, NY	(516) 561 6590	*24
CBBS London, England (European standard)	(044) 1 399 2136	
CBBS Long Island, NY	(516) 334 3134	*24
CBBS MAUDE Milwaukee, WI	(414) 241 8364	*24
CBBS MicroStar, Worcester, MA	(617) 752 7284	
CBBS NW, Portland, OR	(503) 646 5510	
CBBS PACC, Pittsburgh, PA	(412) 822 7176	*24
CBBS Prince George, B.C., Canada	(604) 562 9515	
CBBS Proxima, Berkeley, CA	(415) 357 1130	
CBBS RAMS, Rochester, NY	(716) 244 9531	
CBBS Richfield, MN	(612) 423 5016	
CBBS Strictly Software, Honolulu, HI	(808) 944 0562	
CBBS TSG, Tucson, AZ	(602) 746 3956	*24

COMNET-80 Akron, OH	&(216) 645 0827	*24
COMNET-80 Las Vegas, NV	&(702) 870 9986	
COMNET-80 Mt. Clemens, MI	&(313) 465 9531	
COMNET-80 North Wales, PA	(215) 855 3809	
COMNET-80 Riverside, CA	&(714) 359 3189	
COMNET-80 Riverside, CA	&(714) 877 2253	
COMNET-80 Wichita Falls, TX	(817) 767 5847	

CONNECTION-80 Centereach, NY	(516) 588 5836	
CONNECTION-80 Denver, CO	(303) 690 4566	*24
CONNECTION-80 Escondido, CA	(619) 746 6265	
CONNECTION-80 Fremont, CA	(415) 651 4147	*24
CONNECTION-80 Gaithersburg, MD	(301) 840 8588	*24
CONNECTION-80 Great Neck, NY	(516) 482 8491	*24
CONNECTION-80 Lansing, MI	(517) 339 3367	
CONNECTION-80 Laval BELE, Laval, Quebec, CN	(514) 622 1274	*24
CONNECTION-80 Little Rock, AS	(501) 372 0576	
CONNECTION-80 Lincroft, NJ	(201) 842 7644	
CONNECTION-80 Manhattan, NY	(212) 991 1664	
CONNECTION-80 North York, Ontario	(416) 667 9981	
CONNECTION-80 Orlando, FL	(305) 644 8327	*24
CONNECTION-80 PAUG, Portland, OR	(503) 281 7653	
CONNECTION-80 Peterborough, NH	(603) 924 7920	
CONNECTION-80 Tulsa, OK	(918) 747 1310	*24
CONNECTION-80 W. Mich. Micro Group, MI	(616) 457 1840	*24
CONNECTION-80 Willowdale, Ontario	(416) 226 9260	
CONNECTION-80 Winter Garden, FL	(305) 894 1886	*24
CONNECTION-80 Woodhaven, NY	(212) 441 3755	*24
CONNECTION-80 Tampa, FL	(813) 977 0989	

CONFERENCE-TREE #3, Hayward, CA	(415) 538 3580	
CONFERENCE-TREE #4, Santa Monica, CA	(213) 394 1505	
CONFERENCE-TREE Anchorage, AK	(907) 344 5251	
CONFERENCE-TREE Computerland, Honolulu, HI	(808) 487 2001	*24
CONFERENCE-TREE Flagship, Denville, NJ	(201) 627 5151	*24
CONFERENCE-TREE Kelp Bed, Los Angeles, CA	(213) 372 4800	
CONFERENCE-TREE Minneapolis, MN	(612) 854 9691	
CONFERENCE-TREE ?, New Jersey	(201) 627 5151	
CONFERENCE-TREE Victoria, TX	(512) 578 5833	

DIAL-YOUR-MATCH #1	(213) 842 3322	-so
DIAL-YOUR-MATCH #3	(912) 233 0863	-so
DIAL-YOUR-MATCH #4	(213) 783 2305	-so
DIAL-YOUR-MATCH # 9	(213) 345 1047	-so
DIAL-YOUR-MATCH #11	(213) 242 1882	-so
DIAL-YOUR-MATCH #12, Houston, TX	(713) 556 1531	-so
DIAL-YOUR-MATCH #14	(201) 272 3686	-so
DIAL-YOUR-MATCH #16	(206) 256 6624	-so
DIAL-YOUR-MATCH #17	(415) 991 4911	-so
DIAL-YOUR-MATCH #18	(617) 334 6369	-so
DIAL-YOUR-MATCH #20	(919) 362 0676	-so
DIAL-YOUR-MATCH ##22	(213) 990 6830	-so
DIAL-YOUR-MATCH #23, Omaha, NE	(402) 571 8942	-so
DIAL-YOUR-MATCH #24, Houston, TX	(713) 783 4136	-so
DIAL-YOUR-MATCH #26, Clovis, CA	(209) 298 1328	-so

Complements of Peoples' Message System, Santee CA
(619) 561-7277

(Continued on page 123)

APPLE, Mountain, and Data Capture

by H. Bruce Land, III

The CPS Multifunction Card from Mountain Computer ($180 to $240) and Data Capture 4.0/80, CPS Version ($90) from Southeastern Software, can provide your Apple Computer with a complete RS-232 I/O port with five true handshake lines, a parallel printer output port, clock and calendar, and battery backup. You can have a smart terminal, hardware, and software by using only one of the Apple's few slots and some of its limited power supply — all for as little as $270.

The Hardware

The Mountain Computer card ROM contains a system configuration program that allows you to determine how the card will function. From a menu you can set baud rate, parity, number of data bits, and number of stop bits. You can even use a 5-bit ASCII; and you can set the appropriate functions for the parallel port.

By choosing items from the menu, you can select which slot the Apple "thinks" the card is in, regardless of its actual location. For example, you can place the card in Slot 2 but address the printer as Slot 1, the modem as Slot 3, and the clock as Slot 4. These assignments can be reset by software so that other real cards can reside in these slots.

The serial output can automatically change lower case to upper case if you don't have a lower-case adapter for your Apple, and it can echo characters back to the sender. The serial output also can define a control character to function as an escape character, set or clear the high-order bit, supply auto-line feeds, set line length, do automatic paging for pages of any size, and add a carriage-return delay.

Once you configure the system, the parameters are stored in the permanent CMOS memory and you can ignore them until you want to make changes.

Additional high-level software supplied with the unit allows you to turn the Apple monitor into an analog clock complete with sweep second hand. The CPS Lister program allows you to make formatted Applesoft program listings, properly spaced, with the date and time printed at the top of each page, page numbers, and with no printing over the perforations on continuous paper. If you often forget which is your most recent listing, then these dated and timed listings are for you.

The Software

When you use Applesoft with a normal serial I/O card and type data to a modem, every time you hit RETURN Applesoft says 'SYNTAX ERROR' because it thinks you're erroneously entering a BASIC statement. Install the Mountain CPS card, enter a couple of control codes, and your Apple will function as a dumb terminal. You can talk to another computer through your keyboard, and it can display messages on your CRT. Although you can communicate, at this level of operation you can't send a message to the printer, store it in memory, or save it on disk.

Enter Data Capture 4.0, 40/80 column, CPS version. This combination is not the only one available, but it's the only one I've found that does the whole job. With the CPS card and Data Capture, your Apple can be a computer one moment and a smart terminal the next. You can compose your message off-line and then burst it over the line at up to 1200 baud to another computer, a time-sharing system, or even to a mainframe computer. You can hold a received message in memory, edit it, save it to disk or cassette, and print it at your leisure.

Run Data Capture and press ESC. You will see the following menu:

C)atalog	disk	
D)elete	text	
I)nsert	text	
L)ist	text	
M)erge	from file	
P)rint	text	
Q)uit	program	
S)end	text	
T)oggle:		
		A)lternative drive (1/2)
		B)aud rate
		C)apture (on/off)
		D)uplex (full/half)
		L)ocal carrier (on/off)
		S)pecial characters (on/off)
W)rite to file		

Any of these functions can be selected and executed while you are off-line. When you are on-line you can send a signal to place the other computer on "hold," select and execute commands from the menu, and then resume communication on the other computer. While the computers are talking to each other, status lines display the operating mode and tell you how many lines of text have passed through your Apple.

If the capture mode is off, nothing is saved; if it is on, both sides of the transmission are saved as a text file in a RAM buffer. At any time you can write the text to a disk file for later use; and at any time the buffer can be partially or fully cleared (deleted), relisted, saved, or printed. Additional text can be merged from disk to buffer and then sent to the other computer. You can send and receive text, numerical data,

and program listings, and you can transfer programs directly to another Apple. (*Note:* Data Capture does not work in auto-dial or auto-answer mode with the CPS card.)

Some mainframes require special key codes that the Apple normally cannot generate without Data Capture (for example, the UNIX system I've been using requires a true delete code and an underline). The Apple keyboard generates a backspace and the hardware interprets this as a backspace/delete. Data Capture allows you to redefine portions of the Apple keyboard to generate any ASCII code you may need, including any of the control codes.

Both the CPS card and Data Capture come with more documentation than you'll ever read, but it is comforting to know that it's there in case you want to do something different. My printer is a Selectric typewriter and my modem is homemade, so I needed the extra documentation.

You may wonder why Data Capture is so expensive. The task it must perform is tricky. The Apple cannot talk to two I/O devices at the same time. It cannot send data from the keyboard to both the modem and the display at the same time. Data Capture has so much to do in so short a period of time that it uses machine code for an intricate routine that 1. looks at the keyboard; 2. if data is available there, checks to see whether or not the data is a control character; 3. if not, stores the data in a RAM buffer; 4. sends it to the display; and 5. sends it to the modem. While this is happening, Data Capture must, in effect, look over its shoulder and check the modem to see if it is sending a character to the Apple, decide whether or not this is a control character, and if not, store the character in the buffer and send it to the display.

Meanwhile Data Capture must format each character into the proper word length, control stop bits, baud rate, etc. — all on data moving at speeds up to 1200 baud.

Unlike the high-speed software that handles the bits and the bytes, the software that services the menu is in Applesoft, and you can modify it without difficulty. To get the attention of a big system running under UNIX, I had to change the length of the BREAK command and make it repeat twice. This was easy to do in BASIC.

Data Capture is not copy protected, so if you want to talk to several different systems with different requirements, you can prepare a disk for each, and avoid frequent software modifications.

To sum up, the Apple and the CPS card and Data Capture make a fine team. Together, they can handle anything at 1200 baud or less, and they do it in a friendly fashion.

Note: You can buy the CPS Multifunction Card from Mountain Computer, Inc., 300 El Pueblo Road, Scotts Valley, CA 95066, (408) 438-6650. Data Capture 4.0/80, CPS Version, can be obtained from Southeastern Software, 6414 Derbyshire Drive, New Orleans, LA 70126, (504) 246-8438.

You may contact the author at 6916 Park Place, Baltimore, MD 21227.

MICRO

Alspa Computer, Inc.

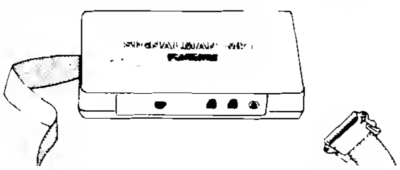
The price-performance leader. Includes Z80A, 1 or 2 full 8" drives (double density, double sided), 3 serial and 1 parallel port, and winchester port. Prices start at less than $2000. DEALER and OEM inquiries invited.

SPECIALS on INTREGATED CIRCUITS

6502	7.45	10/ 6.95	50/ 6.55	100/ 6.15
6502A/6512A	8.40	10/ 7.95	50/ 7.35	100/ 6.90
6520 PIA	5.15	10/ 4.90	50/ 4.45	100/ 4.15
6522 VIA	6.45	10/ 6.10	50/ 5.75	100/ 5.45
6532	7.90	10/ 7.40	50/ 7.00	100/ 6.60
2114-L200		2.45	25/ 2.30	100/ 2.15
2716 EPROM		4.90	5/ 4.50	10/ 4.00
2532 EPROM		7.90	5/ 7.45	10/ 6.90
6116 2KX8 CMOS RAM		7.90	5/ 7.45	10/ 6.90
4116 RAM				8 for 14
Zero Insertion Force 24 pin Socket (Scanbe)				2.00

Hewlett Packard

Write or call for prices.



Anchor Automation Signalman Modems

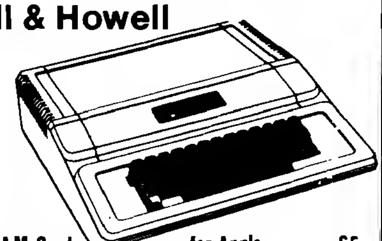
FREE SOURCE MEMBERSHIP WITH SIGNALMAN

All Signalman Modems are Direct Connect, and include cables to connect to your computer and to the telephone. Signalman Modems provide the best price-performance values, and start at less than $100. **Dealer and OEM inquiries invited**

Mark I RS232
Mark II for Atari 850
Mark IV for CBM/PET with software
Mark V for Osborne (software available)
Mark VI for IBM Personal Computer
Mark VII Auto Dial/Auto Answer
Mark VIII Bell 212 Auto Dial/Answer

DC HAYES Smartmodem	**229**
DC Hayes Smartmodem 1200	545

We carry Apple II+ from Bell & Howell



16K RAM Card for Apple	**65**
Solid Oak 2 Level Stand for Apple	**29**
Apple LOGO	150
Video Recorder Interface	545
Super Serial Card	149
Thunderclock Plus	119
Z80 Softcard and CP/M (Microsoft)	235
Parallel Printer Interface/Cable	80
Grappler Interface	139
TG Products Joystick for Apple	**48**
TG Paddles	32
DC Hayes Micromodem II	299
Videx 80 Column Card	239
Hayden Software for Apple 20% OFF	
Apple PASCAL Language	195
Apple FORTRAN	160
We stock EDUWARE Software	
GENIS I Courseware Development System	90
Unicom Grade Reporting or School Inventory	250
Executive Briefing System with fonts	225
Apple Dumpling (Microtek) Printer Interface	**115**
Apple Dumpling with 16K Buffer	**160**
PIE Writer Word Processor	**120**

commodore

See us for Personal, Business, and Educational requirements. Educational Discounts available.

PETSCAN $245 base price

Allows you to connect up to 35 CBM/PET Computers to shared disk drives and printers. Completely transparent to the user. Perfect for schools or multiple word processing configurations. Base configuration supports 2 computers. Additional computer hookups $100 each.

Commodore COMMUNICATES!

COMPACK $115

Intelligent Terminal Package includes:
ACIA hardware based interface; DB25 Cable and STCP Software with remote telemetry, transfer to/from disk, printer output, XON-XOFF control, user program control, and status line.

VE-2 IEEE to Parallel Interface 110

Includes case, power supply, full 8-bit transmission, and switch selectable character conversion to ASCII.

VIC 20 Products		VIC Sargon II Chess	32
Backup V1.0	20	VIC GORF	32
VIC RAM Cards in stock		Meteor Run (UMI)	39
VIC SuperExpander	53	VIC Radar Ratrace	24
VIC 16K RAM	95	Amok (UMI)	20
Thorn EMI Software		Snakman	15
HES Software		Rubik's Cube	13
VIC Omega Race	32	Programmers Reference	15
Spiders of Mars (UMI)	39	Renaissance (UMI)	39
Programmers Aid	45	VIC Adventure Series	

VICTORY Software for VIC and C64

Street Sweepers	12	Maze in 3-D	12
Night Rider	11	Cosmic Debris	12
Treasures of Bat Cave	12	Grave Robbers Advent.	11
Games Pack I	12	Games Pack II	12
Victory Casino	8	Adventure Pack I	12
Adventure Pack II	12	Trek	11

Commodore 64 Programmers Reference Guide	17
Compute!'s First Book of PET/CBM	11
POWER ROM Utilities for PET/CBM	78
WordPro 3+ - 32K CBM, disk, printer	195
WordPro 3+/64	**70**
WordPro 4+ - 8032, disk, printer	300
SPELLMASTER spelling checker for WordPro	**170**
VISICALC for PET, ATARI, or Apple	190
PETRAX PET to Epson Graphics Software	**40**
SM-KIT enhanced PET/CBM ROM Utilities	**40**
Programmers Toolkit - PET ROM Utilities	35
PET Spacemaker II ROM Switch	36
2 Meter PET to IEEE or IEEE to IEEE Cable	40
Dust Cover for PET, CBM, 4040, or 8050	8
VIC or C64 Parallel Printer Interface	79
CmC IEEE-RS232 Printer Interface — PET	120
SADI Intelligent IEEE-RS232 or parallel	235
ZRAM - CBM 64K RAM, Z80, CP/M	550
Programming the PET/CBM (Compute!) — R. West	**20**
Compute! First Book of VIC	11
Whole PET Catalog (Midnight Gazette)	**8**
Color Chart Video Board for PET	125
PET Fun and Games (Cursor)	11

FlexFile for PET CBM, C64 $110

Database, Report Writer with Calculations, Mailing Lists

FORTH for PET full FIG model — Cargill/Riley	**$50**
Metacompiler for FORTH for independent object code	**30**
KMMM PASCAL for PET/CBM/C64	**85**
EARL for PET/CBM Disk-based ASSEMBLER	**65**
Super Graphics — BASIC Language Exercises	**45**

Fast machine language graphics routines for PET/CBM

RAM/ROM for PET/CBM 4K $75 8K $90

DISK SPECIALS

Scotch (3M) 5" ss/dd	10/ 2.25	50/ 2.10	100/ 2.05
Scotch (3M) 5" ds/dd	10/ 3.15	50/ 2.90	100/ 2.85
Scotch (3M) 8" ss/sd	10/ 2.40	50/ 2.20	100/ 2.15
Scotch (3M) 8" ss/dd	10/ 2.95	50/ 2.70	100/ 2.65

We stock VERBATIM DISKS

Write for Dealer and OEM prices.

BASF 5" or 8"	10/ 2.00	20/ 1.95	100/ 1.85
NEW BASF Qualimetric Disks also in stock.			
Wabash 5" ss/sd	10/ 1.80	50/ 1.75	100/ 1.70
Wabash 5" ss/dd	10/ 2.00	50/ 1.95	100/ 1.90
Wabash 8" ss/sd	10/ 2.00	50/ 1.95	100/ 1.90

We stock MAXELL DISKS

Write for dealer and OEM prices.

Disk Storage Pages	10 for $5	Hub Rings 50 for $6
Disk Library Cases	8"—3.00	5"—2.25
Head Cleaning Kits	11	

CASSETTES—AGFA PE-611 PREMIUM

High output, low noise, 5 screw housings.

C-10	10/ .61	50/ .58	100/ .50
C-30	10/ .85	50/ .82	100/ .70

SPECIALS

Zenith ZVM-121 Green Phosphor Monitor	109
VOICE BOX Speech Synthesizer (Apple or Atari)	
Many printers available (Star, Brother, OKI, etc.)	
We Stock AMDEK Monitors	
Watanabe Intelligent Plotter 990 6-pen	1290
ISOBAR 4 Outlet Surge Supressor/Noise Filter	**49**
We stock Electrohome Monitors	
dBASE II	390
Panasonic TR-120M1P 12" Monitor (20 MHz)	149
Panasonic CT-160 Dual Mode Color Monitor	285
Franklin Computers - special system price	
Hewlett Packard Calculators available	

USI Video Monitors—Green or AMBER 20 MHz hi-res. Dealer and OEM inquiries invited

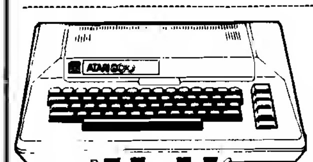
ALL BOOK and SOFTWARE PRICES DISCOUNTED

A P Products	**15% OFF**
Synertek SYM-1 Microcomputer	**SALE 189**
KTM-2/80 Synertek Video and Keyboard	349

ZENITH data systems

Z29 Terminal (VT100, VT-52, ADM3A, Hazl500 compatible)	**680**
ZT-1 Intelligent Communications Terminal	**479**
Z100 16-bit/8-bit System	**CALL**

We stock entire Zenith line.



ATARI SPECIALS

800 Computer	499	Microsoft BASIC	72
400—16K	199	**MISSILE COMMAND**	**29**
810 Disk Drive	440	**ASTEROIDS**	**29**
Thorn EMI Software		**STAR RAIDERS**	**34**
850 Interface	170	Space Invaders	29
Inside Atari DOS	18	**Atari Graph.** (Compute!)	11
Joysticks or Paddles	19	Caverns of Mars	33
Microtek RAM Cards		**PAC-MAN**	**36**
EduFun Software		**CENTIPEDE**	**36**
Pilot	65	First Book of Atari	11
Super Breakout	29	**Anchor Modem—Atari**	**85**
APX Software	Call	Other Atari products	Call

252 Bethlehem Pike
Colmar, PA 18915

215-822-7727

A B Computers

WRITE FOR CATALOG

Add $1 25 per order for shipping We pay balance of UPS surface charges on all prepaid orders Prices listed are on cash discount basis Regular prices slightly higher Prices subject to change

Circle No. 29

Unleash the AIM "A" Block

by Tom Lillevig

Memory is a valuable commodity on the AIM 65. This article shows how to recover some memory space and provides suggestions for uses.

Figure 1: **AIM-65 modification.**

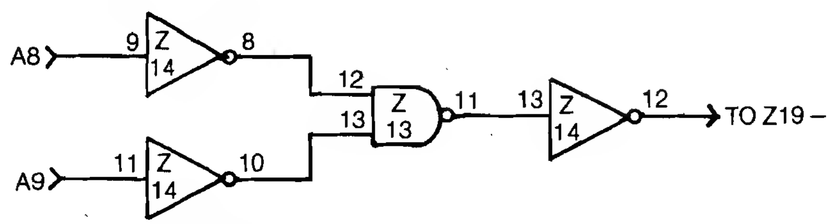
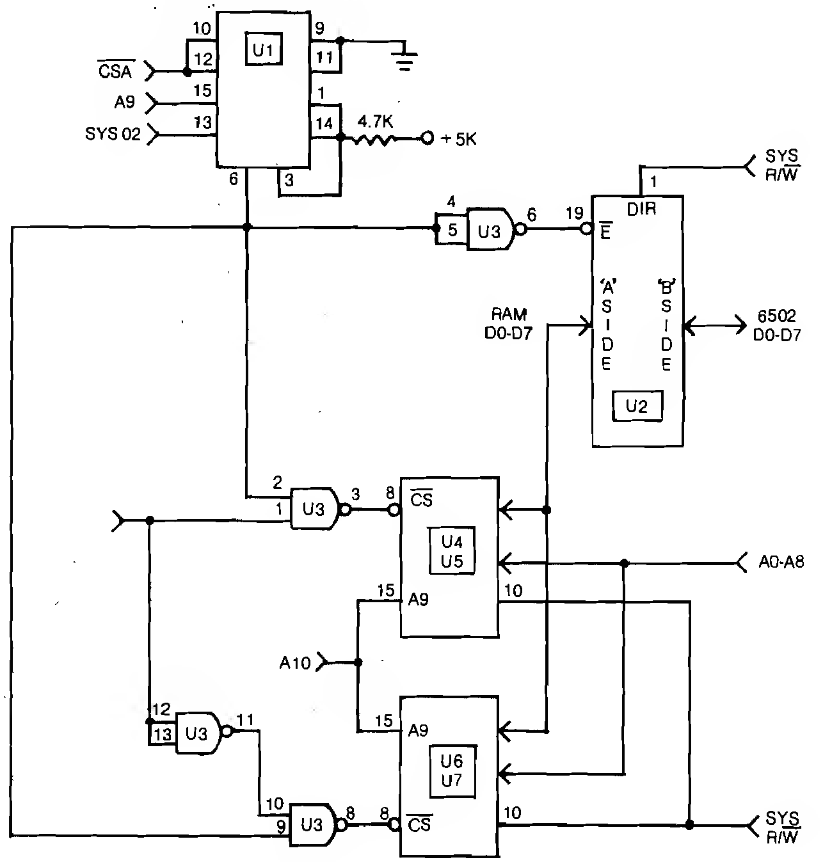


Figure 2: **Adding 2114 RAM to "A" block.**



U1:7485
U2:74LS245
U3:74LS00
U4-U7:2114 RAM

	ADDRESSES
U4,U5:	AA00-ABFF
	AE00-AFFF
U6,U7:	A200-A3FF
	A600-A7FF

Your first look at the address map for the AIM 65 reveals a 36K block of space available for those helpful additions every computer user needs. If, however, you add a couple of 16K RAM boards and a video interface, you soon discover that 36K isn't as much space as you thought.

If you take a closer look at the address map you will see a design that saved Rockwell some money in manufacturing, but cost you the use of valuable memory space. Four interface devices, which require a total of less than 256 bytes of memory, have been allotted an entire 4K block! I refer, of course, to the PIA, RIOT, and VIAs that inhabit the "A" block. This article discusses a simple method to unleash much of the "A" block and several applications for the available space.

The reason that the four devices in the "A" block take up so much memory is that the enable signals are produced by loose decoding. The AIM 65 schematic shows that the enables come from decoder Z19 and are derived from CSA and address lines A10 and A11. This method of decoding allocates 1K of memory space to each device. A better method, first proposed by Larry

Westergren in the computer club newsletter *Interchange*, squeezes each device into a space of 256 bytes, thus freeing up 3K of usable memory. Larry's method requires the addition of one IC, so I decided to see if the decoding could be done using spare gates on the AIM.

See figure 1 for my update of Larry's idea. The NAND gate and inverters are all spare devices, and no circuit cuts are required. The connection to Z19-1 deactivates the existing "A" block enables to the I/O devices, except when both A8 and A9 are at zero. This modification makes address blocks $A100-$A3FF, $A500-$A7FF, $A900-$ABFF, and $AD00-$AFFF available.

Now that these blocks are free, what can be installed? Since each slot is only 768 bytes wide, RAM addition does not appear to be a good choice. If, however, you can live with four separate blocks of 512 bytes, then you can wire four 2114's to provide 2K of memory with no waste or overlap (see figure 2). The RAM blocks occupy addresses $A200-$A3FF, $A600-$A7FF, $AA00-$ABFF, and $AE00-$AFFF. As shown in figure 3, a 6116 RAM or 2716 EPROM could be installed instead of the 2114's.

The rest of the available space can be decoded further to provide enable lines for a variety of devices. The circuit in figure 4 illustrates a simple method for deriving eight enables from the remaining blocks. The enables may be used for any chips that require 128 bytes of memory space, or less. PIAs, VIAs, and real-time clocks are just a few example of devices that will fit nicely.

Tom Lillevig is a Senior Training Representative at Rockwell-Collins. He is also secretary of the Cedar Valley Computer Association, an organization that includes nearly 500 AIM 65 owners. You may contact Mr. Lillevig at 130 Carnaby Dr. NE, Cedar Rapids, IA 52402.

MICRO

Figure 3: Adding 6116 RAM or 2716 EPROM to "A" block.

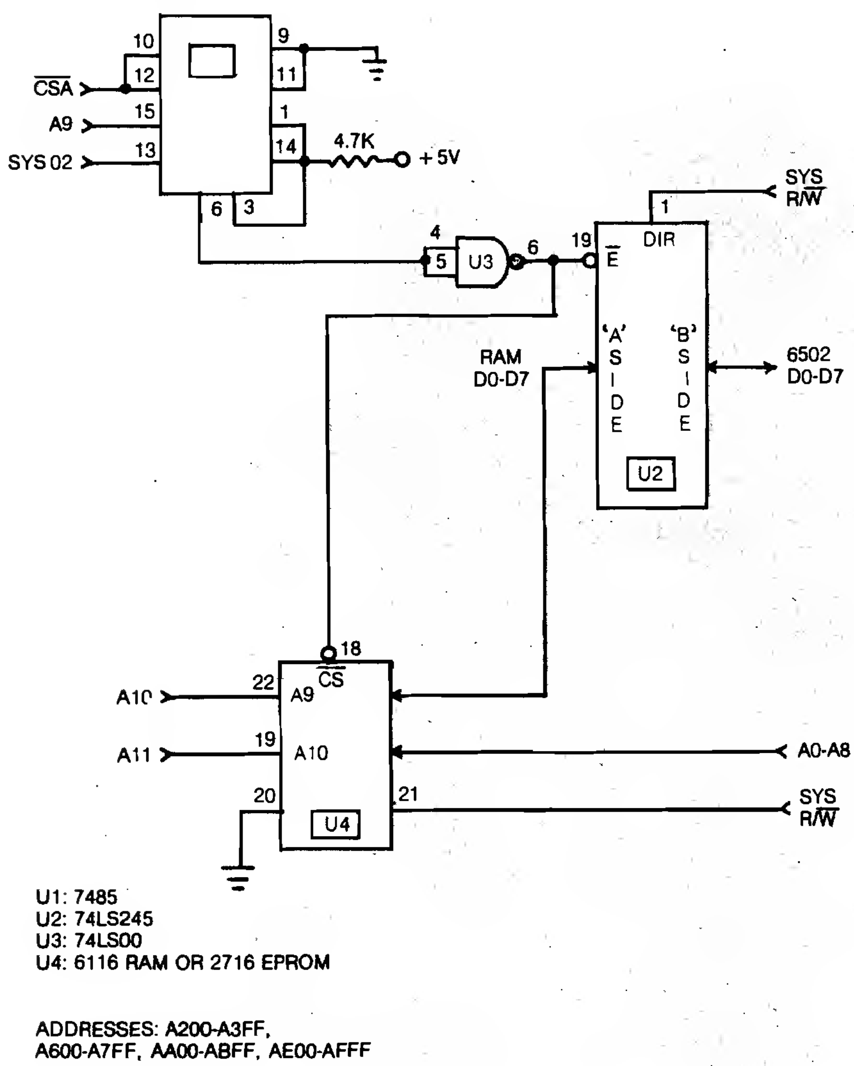
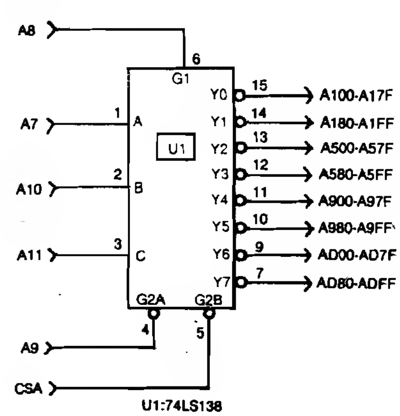
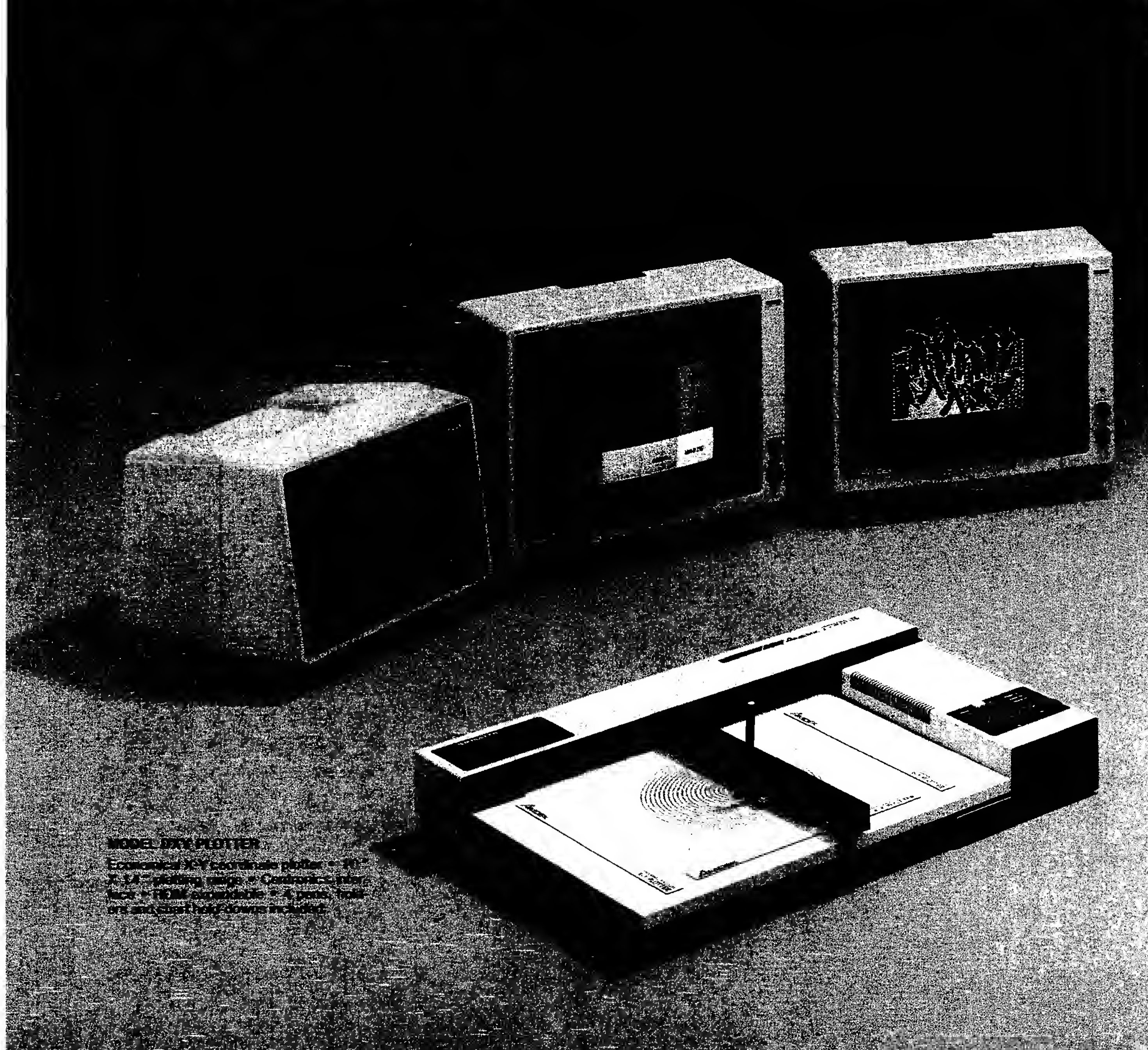


Figure 4: Decoding for spare enable lines.



AMDEK...your guide to



MODEL DXY PLOTTER
Economical X-Y coordinate plotter • 10" x 14" plotting range • Centronics interface • [illegible] compatible • 4 pens, holders and chart hold-downs included.

YOUR COMPATIBILITY CHART

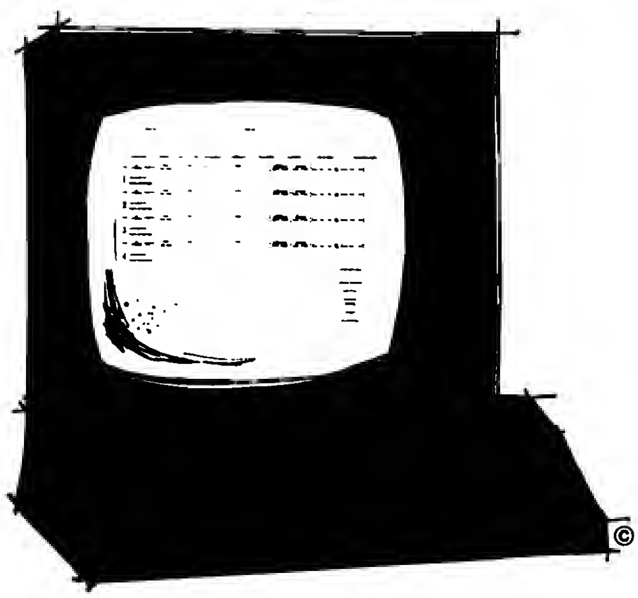
MORE COMPATIBILITY INTERFACES DUE SOON . . . CHECK FACTOR

COMPUTER	MONITORS							PLOTTER	3" MICRO-DISC DRIVE	NOTES
	VIDEO-300	VIDEO-310	COLOR-I	COLOR-II	COLOR-IIA	COLOR-III	COLOR-IV			
IBM-PC	●	●	●	●	*	*	*	●	●	* Special Cabling Required
APPLE III	●			*	*	*	*	●		* Special Cabling or Converter Required
APPLE II	●		●	*	*	*	●	●		* DVM Board Required
ATARI 800	●		*							* Opt. Atari Cable Required
VIC-20			●							* Opt. VIC Cable Required
TRS-80	*									* Opt. TRS Cable Required
Osborne	●									* Opt. Interface Required
TI-99			●							* Opt. TI Cable Required
Commodore-64	*		*							* Opt. Commodore Cable Req.

The MICRO™ LEARNING CENTER

Atari 800	Texas Instrument

Features:



- **MASTER for VIC and COMMODORE 64**
  by Loren Wright

  A guessing game involving concealed letter patterns.

- **Conservation of Momemtum for Atari and Commodore**
  by Terry Faughn

  A tutorial appropriate for an introductory physics class.

- **Is a Number a Number?**
  by Phil Daley

  An easy way to convert from one base to another.

- **A Beginner's Computer Glossary**

  Part two of our introductory dictionary.

MASTER for VIC-20 and COMMODORE 64

by Loren Wrigh

Apple Listing appears on page 82

MASTER is a simple guessing game for one or two players. The commercial version of this game involves colored pegs. One player constructs a pattern of four colored pegs behind a screen, and it is up to the other player to guess the concealed pattern. The first player provides the second player with clues, telling him how many pegs have been guessed in the right position, and how many pegs are the right color but in the wrong position. The second player continues to guess until he has discovered the colors and correct positions of all four pegs. The number of guesses is the score, and the player with the lower score wins. The computer uses letters instead of pegs, but the rules are the same. In fact, the MASTER program offers you a choice of three different game versions, and you can modify the program to play even more games.

Running the Program

Position the tape to load the program MASTER. Hold down the shift key and press RUN/STOP. When the program is loaded, the screen will clear and the message '1 OR 2 PLAYERS?' will be displayed at the top. For the moment, select '1'. The two-player game is described later. Next you are offered a menu of game difficulty levels. Press '1', '2', or '3' to select a game. (You can change your choice for the next game, if you want.) The rules appropriate to the game you have selected are then displayed. The rules are printed here for reference.

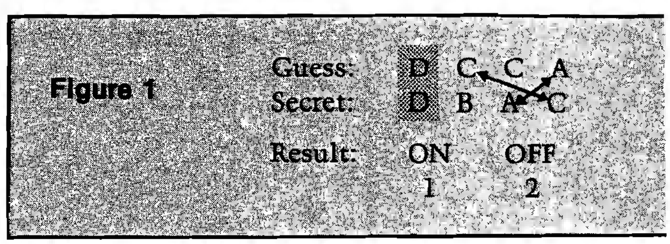
In the EASY game, only A, B, C, and D are allowed, and no letter may be repeated in the secret pattern. Your guesses may include repeated letters, though. In the MID game, only A, B, C, and D are allowed, but these letters may be repeated in the pattern. In the HARD game, A, B, C, D, E, and F are allowed, and letters may be repeated.

Press any key (except RUN/STOP) to continue. The computer now generates, at random, a secret pattern. The screen will clear and appear as below:

```
SELECT LETTER ON OFF
 > ■
```

The flashing-square cursor appears right after the '>'. Only '?', '←', and the letters allowed in the game will be accepted from the keyboard. (The RUN/STOP key does work, though!) Acceptable characters will be printed on the screen; unacceptable ones will have no effect. As soon as you enter the fourth letter in your guess pattern, the program will process it. Until you enter that fourth character, though, you may change your mind. Press '←' to restart your guess. If at any time you want to give up, the '?' key will print the secret pattern and let you start over with a new game and pattern.

When you enter the fourth item in your guess pattern, the computer matches it against the secret pattern. In the 'ON' column is the number of letters guessed correctly, and in the right position; in the 'OFF' column is the number guessed right, but in the wrong position. Understanding the matching process will help you learn to play the game better. For instance, if you guessed 'D C C A', and the secret pattern is 'D B A C', the computer will return a '1' in the 'ON' column and a '2' in the 'OFF' column.



The 'D' is in the correct position (indicated by the shading), but the 'C' and the 'A' (matches indicated with arrows), while they do exist in the

secret pattern, were guessed in the wrong position. Only one of the C's in the guess is counted, since there is only one C in the secret pattern. If the secret pattern were 'C B A C' instead, the program would return '0' in the 'ON' column and '3' in the 'OFF' column. Both C's in the guess are now counted.

Figure 2

Guess:	D	C	C	A
Secret:	C	B	A	C
Result:	ON		OFF	
	0		3	

When you have guessed the secret pattern correctly, you will be congratulated and told the number of guesses you took. Then the program is restarted with selection of the game level.

As you play more and more games, you will begin to develop systems to help you guess the pattern as quickly as possible. One technique that is sometimes useful is substituting one character at a time.

Two-person Game

The two-person option allows a second player to input a secret pattern instead of having the computer come up with one. The player who will be guessing should look away from the screen while the other player inputs a pattern. The program tests for the letters allowed but it does not check for repetitions. Be sure to follow the repetition rule in effect. To go back to the one-person version press RUN/STOP, type RUN and press RETURN. This time answer '1' for the number of players.

Programming Techniques in MASTER

Random Numbers

In the one-player version of MASTER, the program is able to come up with a different secret pattern of letters each time the game is played. How is this done? The secret is in random numbers. BASIC is able to generate random numbers using the RND function.

A random number is one that is obtained without any predictability or repeatability. Rolling a die, flipping a coin, and spinning a roulette wheel are all means of obtaining random numbers in the real world.

Many programming applications require a source of random numbers. For statistics programs they can provide sample data to test a model, and in physics they can be used for applications such as demonstrating the behavior of gas particles.

Many game programs require random numbers. These may be used in the form of playing cards, dice, or locations of hidden treasures. In the 1-player version of MASTER, random numbers are used to generate the secret pattern of letters.

The BASIC function RND generates pseudo-random numbers in the range between 0 and 1. Pseudo-random means each succeeding number depends to some extent on the previous one. As a result, after many thousands of numbers, the sequence will start over. This makes statistics involving very large samples sometimes difficult, but it usually causes no problem in games, which use considerably fewer numbers.

The only problem we must avoid is generating the same sequence of random numbers every time the program is run. The technique used in MASTER is to be sure the RND function has a different starting number or 'seed' each time. This is accomplished with the statement 'I=RND(-TI)' in line 8040 as part of the initialization sequence. Using a negative number as the argument for RND causes a function of the argument to be used as the seed to start the sequence of random numbers. TI is the value of an internal clock that starts at zero when the computer is turned on and increments every sixtieth of a second. Since you are very unlikely to start the program at exactly the same moment each time, you are practically assured of getting a different seed each time. If you use a negative constant instead of -TI in line 8040, you will generate the same sequence of random numbers each time. Run the program this way and you will be able to astound your friends with your psychic powers!

Now that we have a sequence of random numbers, how do we turn this sequence into the letter patterns for MASTER? Line 1020 does it all in one BASIC expression: RN=INT (RND(1)*N+1), where N is number of letters allowed in the game. See figure 3 for a graphic illustration of how four random numbers are converted into the four letters of a secret MASTER pattern. RND(1) produces numbers in the range of 0 to 1, but this does not include either 0 or 1 themselves. First we multiply the number by the number of letters allowed in the game. If we allow four letters (N=4), then we multiply the random numbers by four to get numbers in the range 0 to 3.999.... Next we add 1 to make it 1 to 4.999.... Then we use the BASIC INT function to remove whatever is to the right of the decimal point, leaving us with 1, 2, 3, or 4. These numbers are never actually converted to letters. Instead, the letters the player types for a guess are converted to numbers.

Figure 3

BASIC Function	Element # 1	2	3	4
RND(1)	.555877482	.689094948	.828839479	.0619696133
*4	2.22350993	2.75637979	3.31535792	.247878453
+1	3.22350993	3.75637979	4.31535792	1.24787845
INT( )	3	3	4	1
LETTER	C	C	D	A

The same technique can be used to get random numbers over any range. For dice, multiply by six, take the integer, and add 1. For playing cards, multiply by 52, take the integer, and add 1. (Converting 1 to 52 into suits and ranks is another problem!)

Flags and Logic

One of the most powerful features of a computer is its ability to make decisions. MASTER uses the computer's decision-making ability throughout its program.

Every decision boils down to deciding whether an expression is true or false. The BASIC IF...THEN construction decides whether an expression is true or false. If the expression after the IF is true, then whatever appears on the line after the THEN is executed. If the expression is false, then the rest of the line is skipped and execution continues with the next line.

BASIC doesn't actually handle the words 'true' and 'false.' Instead, it assigns −1 to represent 'true' and 0 to represent 'false'. When evaluating expressions, any non-zero result is considered 'true'. To see this in action try the following example:

```
10 INPUT "A=";A
20 INPUT "B=";B
30 IF A=B THEN PRINT "TRUE": GOTO 50
40 PRINT "FALSE"
50 PRINT A=B
60 GOTO 10
```

Run this program. Type in a value for A, press RETURN, type in a value for B, and press RETURN. If the number you entered for A equals the number you typed for B, then 'TRUE' will be printed, followed by −1. Otherwise 'FALSE' is printed, followed by 0. The number −1 or 0 is the value BASIC assigned to the expression 'A=B'. Line 180 in MASTER checks to see if the number of correct position matches (PM) is equal to the number of letters (NN) in the pattern. If so, the player has correctly guessed the pattern and the congratulation routine 6000 is executed before starting a new game by returning to line 100.

Now enter the following program example that demonstrates the use of a flag.

```
10 INPUT A
20 IF A THEN PRINT "TRUE": GOTO 40
30 PRINT "FALSE"
40 GOTO 10
```

Try a few numbers. Every number except 0 will result in 'TRUE' being printed. Entering 0 will produce a 'FALSE'. The 'A' in line 20 is evaluated just like any other expression. If it is non-zero then it is considered true.

A flag is a convenient device in a program. It can be either set (true or −1) or clear (false or 0). BASIC doesn't have a special variable type for flags, but either integer of floating point variables may be used that way. MASTER uses several variables as flags: RP, RQ, and the arrays PF( ) and PG( ). RP is set or cleared in the game selection routine (in line 7100, 7200, or 7300), depending on the game chosen. In line 1030, if RP is set (= −1) then lines 1040-1080, which prevent duplicate letters in the pattern, are skipped. RQ stays cleared unless a duplicate letter is found. If the flag is set, then the program returns to 1020 to determine a new number. Each element of the secret pattern has an element in the flag array PF( ), and each element in the guessed pattern has an element in the flag array PG( ). See the discussion under "Processing a Guess" for details of how these flags are used.

Another interesting use of a flag is in the display of the congratulation message (6080-6150). A FOR...NEXT loop is used to alternate the variable I between −1 and 0. The flag I is tested in lines 6090 and 6120. If the flag is set, then the reverse-field character is printed. When the flag is clear, the following message is printed in normal characters. This produces the alternating reverse-field effect.

The program has to make decisions in a number of other places, evaluating an expression

to determine what to do next. The IF...THEN statement is used most commonly for decision making, but ON...GOSUB and ON...GOTO are also used. ON...GOSUB is used in line 110 to decide whether to generate a random pattern in a 1-player game, or to let a player input a pattern.

Processing a Guess

As explained earlier, the match count is determined by first checking for exact position matches and then going through to check for out-of-position matches. No element in either the secret or guess pattern may be used more than once in a match.

To avoid re-using pattern elements in matches, we need to program a way to "cross off" pattern elements that have been used in a match. In addition to the two arrays of the elements themselves, two corresponding flag arrays are used.

At the beginning of the matching process, all the flags are cleared, or set to zero (3010-3030). As each match is detected, the flags corresponding to the matched elements are set (in lines 3050 and 3550). The flags are checked in lines 3520 and 3540. If the flag is set, then the matching process is skipped and the next element is checked. In addition, when a match is found in line 3550, the higher numbered elements in the guess pattern are skipped by setting the loop index J to its maximum value, NN. The NEXT J statement in line 3560 sees J equal to its maximum value and is fooled into thinking it's through with the specified repetitions. Control passes to the NEXT I statement in line 3570.

This process is graphically demonstrated in figure 4. I is the index into the secret pattern, while J is the index into the guess pattern. The boxes indicate the two elements currently being compared, PM is the number of position matches, and OM is the number of out-of-position matches. A shaded box indicates a match and a diagonal line through an element indicates that it has been used in a match already. First, the position matches are checked. The result is 1, with the D's in the first position crossed off. In the program, the flags PF(1) and PG(1) are set to -1.

Next, the out-of-position matches are checked. Since the first elements in each pattern have already been used, the comparison begins with the second elements. No match is found for the B, so the search continues with the third element of the secret pattern and the second element of the guess pattern. When the match is found with the fourth guess element, these two are crossed off, and the out-of-position match counter OM is incremented. A match is found immediately for the fourth secret pattern element, so the remaining two elements are skipped, and the counter incremented again. *One* position match and *two* out-of-position matches are reported to the player.

If you are still confused about how this works, try a different pattern and construct a table similar to figure 4. You might also try running through the program lines with an example.

Customizing your MASTER Game

Adding an EXPERT Level

Because of the way MASTER is written it is easy to add your own version to the game. As an example of how to do this, let's add an EXPERT game to the three choices we have already. Add or substitute the following lines to the program supplied.

```
2100 PRINT"[RVS]"CHR$(T+64)"[OFF]";
7050 PRINT"[CD][2 CR][RVS]1[OFF]     EXPERT"
```

Figure 4

	Secret Pattern					Guess Pattern					
I					J					PM	OM
										0	0
1	D	B	A	C	1	D	C	C	A	1	0
2	~~D~~	$\boxed{B}$	A	C	2	~~D~~	$\boxed{C}$	C	A	1	0
3	~~D~~	B	$\boxed{A}$	C	3	~~D~~	C	$\boxed{C}$	A	1	0
4	~~D~~	B	A	$\boxed{C}$	4	~~D~~	C	C	$\boxed{A}$	1	0
2	~~D~~	$\boxed{B}$	A	C	2	~~D~~	$\boxed{C}$	C	A	1	0
2	~~D~~	$\boxed{B}$	A	C	3	~~D~~	C	$\boxed{C}$	A	1	0
2	~~D~~	$\boxed{B}$	A	C	4	~~D~~	C	C	$\boxed{A}$	1	0
3	~~D~~	B	$\boxed{A}$	C	2	~~D~~	$\boxed{C}$	C	A	1	0
3	~~D~~	B	$\boxed{A}$	C	3	~~D~~	C	$\boxed{C}$	A	1	0
3	~~D~~	B	A	C	4	~~D~~	C	C	A	1	1
4	~~D~~	B	~~A~~	C	2	~~D~~	C	C	~~A~~	1	2

```
7070 T=VAL(T$):IFT<1ORT>4THEN7060
7080 ONTGOSUB7100,7200,7300,7600
7600 N=8:RP=-1:G(1)=8:G(2)=12:G(3)=16:
     G(4)=20:G(5)=25
7610 PRINT"[CLR]EXPERT GAME:"
7620 GOSUB7400
7630 PRINT"[CD][2 CR]MORE THAN ONCE"
7640 RETURN
```

This version of the game allows the first eight letters of the alphabet. The operation of the game itself is controlled by the values of N and RP in line 7600. The rest of the program changes involve adding the game to the menu and displaying the rules. The value of N determines the number of letters allowed in the game. RP is a flag, which, if set, allows repeats of letters in the pattern (see the "Flags" section above). The array G( ) holds the cut-off numbers of guesses for each congratulation message. Adjust these values and program the appropriate messages, as in the example above, and you will be able to add your own game version.

Congratulation Messages

As part of the initialization routine, six congratulation messages are defined in lines 8060 and 8070. You can change these messages, as well as the cut-off values G( ) defined in lines 7100, 7200, and 7300.

Number of Elements in Pattern

The number of elements is four for all versions of the game described so far. This number can be changed to practically any number, the only limitations being the width of the display and the amount of memory in your VIC-20. The number of elements in the pattern is determined by the value of NN in line 8050 of the initialization routine. Change line 8050 to read: 8050 NN=3. Now run the program. Notice that everything works as before, except only three letters are generated in the secret pattern, and only three are expected in each guess.

To program more elements in the pattern, two additional changes must be made, both in line 8040:

```
8040 FD$="[BLK][SPC]":BK$="[CL]"
     :CR$=CHR$(13):CF=204
```

With this change, five or six elements can be accommodated without disturbing the rest of the display. Substitute for line 8050, as above: 8050 NN=5 or 8050 NN=6. One solution for longer patterns is to print the clues on the next line:

```
180 PRINTTAB(34)"[BLK][SPC]"PM;OM
```

Another solution is to further compress the letters in the guess:

```
8040 FD$="[BLK]":BK$" ":CR$=CHR$(13)
     :CF=204
```

Patterns of 11 or more elements require a DIM statement in the initialization routine. For example,

```
8050 NN=11:DIMR(NN),GU(NN),PF(NN),PG(NN)
```

along with one of the display adjustments above, sets up the game for 11 elements.

Program Description

Initialization (10): Subroutine 8000 sets up a number of constants, and subroutine 7500 gets the number of players.

Program mainline (100-200): Subroutine 7000 gets the skill level for the game and displays the instructions for the game. Subroutine 5000 waits for a key to be pressed before continuing with the main program.

Line 110 uses the ON...GOSUB structure to determine whether to call subroutine 1000, which generates a random pattern, or subroutine 4000, which allows one player to input a pattern. NP can have only two values, 1 or 2. On 1, subroutine 1000 is called; on 2, subroutine 4000 is called.

GN is used to count the number of guesses. Line 130 calls subroutine 2000, which prints the header on the screen and receives the first guess. The second and subsequent guesses return to line 140 where the same subroutine is called at 2020, to avoid having the header reprinted for each guess.

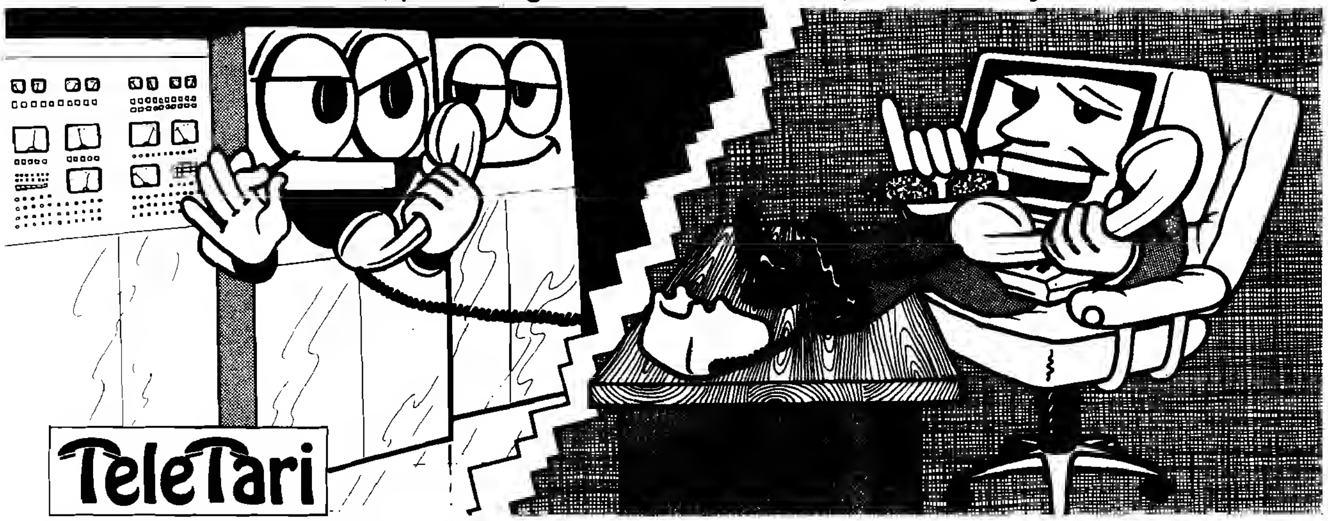
A '?' indicates that the player has given up. A call is made to subroutine 9000, which prints out the secret pattern. GOTO 100 starts the player out with a new game.

Next the guess must be processed. Before each call to the processing routines, the match counters PM and OM are zeroed. Subroutine 3000 processes the guess, first checking for position matches and then for out-of-position matches. If PM (the number of position matches) equals NN (the number of elements in the pattern), then the player has guessed the pattern. Subroutine 6000 is the congratulations routine.

Line 190 prints out the results of the matching, with the position matches under the heading 'ON' and the out-of-position matches under the heading 'OFF'. When the TAB(12) expression is

DON'T ASK PROVIDES THE MISSING LINKS

the link between your modem and the outside world. For hassle-free communications, phone right in with TELETARI, The Friendly Terminal.



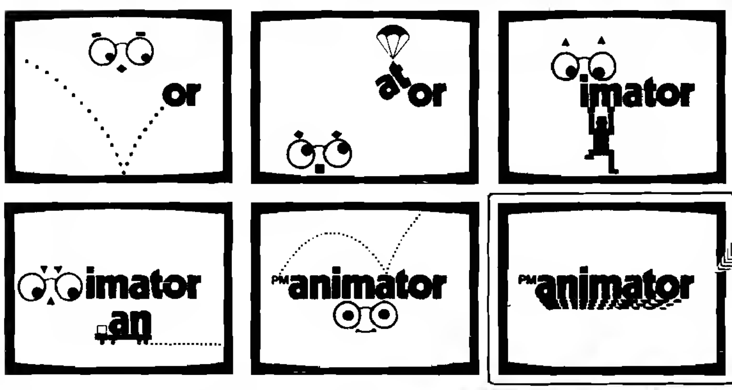
Your Atari has never had such easy access to the whole world of telecommunications – bulletin boards, news reports, large time-sharing computers, the works. Now it's a snap to tap into all these, and it's just as easy to transfer your program or text files to and from a remote computer. Meet TELETARI, The Friendly Terminal. It's just what your modem needs: a powerful, adaptable telecommunications package that's a cinch to use. With TELETARI, you simply choose the desired communications function from a menu. Commonly used terminal parameters are included in the program, but you can change them to suit your needs with a couple of keystrokes, using another handy menu, and store the ones you plan to use again. TELETARI's generous buffer stores up to 20K, so you can review, print, or save received information long after you've hung up the phone. You never knew using a modem could be so convenient. Because it's very flexible, TELETARI is compatible with most modems and a wide variety of computers. And because it works through the RS 232 port, TELETARI is not limited to modem/telephone uses. Put it to work in any RS232 application your imagination can devise – even operating a laser disk!

- buffer of up to 20K
- menu-driven
- highly adaptable
- supports all 850 options
- compatible with 1200 baud modems and BIT 3 Full-view 80* board
- suitable for any RS232 application

$39.95 Requires Basic, 32K RAM, disk, 850 Interface

---

the link between BASIC and arcade-style graphics. Draw and animate pictures for your own BASIC games and other programs with pm ANIMATOR. Create running men, flying rockets, moving figures of all kinds.



Now available from DON'T ASK

BASIC programmers, pm ANIMATOR puts the power of Player-Missile Graphics at your fingertips.

$34.95 Requires 32K RAM, BASIC, disk.

PM animator

To order direct from Don't Ask, send a check or money order, or call to order COD. Add $2.00 for shipping and handling. California residents add 6% sales tax (6.5% if you reside in L.A. County).

---

the link between last game action and verbal learning:

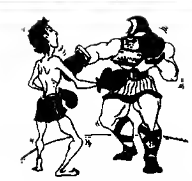
WORDRACE

Kids and adults, increase your vocabulary while you compete in this exciting word game.

Disk version:
3 levels of play – Beginner, Regular, Challenge
Requires 32K RAM, disk, BASIC. $24.95

Cassette version:
2 levels of play – Beginner, Intermediate
Requires 16K RAM, cassette, BASIC. $19.95

turn WORDRACE into a history game or a famous athletes game, and get more vocabulary words, with the WORDRACE accessory disk: CLAIM TO FAME/SPORTS DERBY. 3 new games in all.
Disk only. Requires WORDRACE disk. $19.95



the link between you and what your Atari is really thinking:

ABUSE

the insult-exchange program.
Have you cursed out your computer? Now it can understand you and answer back!
Requires 40K RAM, BASIC, disk. $19.95
Release your aggressions! Inflict ABUSE on anyone who's got it coming!

DON'T ASK
the link between technical excellence and the fun of computing. Why do we give you so much? Don't Ask.

DON'T ASK INC
COMPUTER SOFTWARE

2265 Westwood Bl., Ste. B-150
Los Angeles, CA 90064
(213) 477-4514 or 397-8811

Circle No. 36

encountered in a PRINT statement the cursor moves to the twelfth character position on the screen.

The guess counter GN is incremented and the program loops back to 140 for another guess.

Generate Random Numbers (1000-1110): This routine is called at the beginning of each one-player game to generate the secret pattern. In the supplied version of the game NN is always 4, so four numbers are generated. Line 1020 returns in RN an integer between 1 and the number of letters allowed in the game (N). If RP is non-zero, then repeats are allowed in the pattern. Lines 1040-1080 are skipped and RN is copied into R(I), the current element of the pattern. If repeats are not allowed (RP = 0), then each RN must be checked against the previous elements in the pattern R( ). In line 1040 RQ is set to 0 to indicate that no element has been found so far to match RN. If I = 1 then there aren't any numbers in the pattern and we can skip to 1090 and accept this RN. The FOR...NEXT loop on J (lines 1050-1070) goes from 1 to the previous element (I – 1). If RN is found to match an existing element (RN = R(J)) then RQ is set to – 1 to indicate a match has been found and J is set to I – 1 to terminate the FOR...NEXT loop. If no match is found, then the loop continues through all the previously assigned elements. RQ is tested in line 1080: if it is non-zero, then another RN must be calculated (return to 1020); if it is still zero, then we can accept the RN and install it in the current element R(I) of the pattern. The outside FOR...NEXT loop (1010 to 1100) continues until all of the elements required in the pattern have been calculated.

Process Guess (2000-2130): As discussed above under the program mainline, this routine is usually called at 2020, but the first time the call is made to 2000 to print the heading 'SELECT LETTER ON OFF'.

The routine consists of a big FOR...NEXT loop, where I starts with a value of 1 and ends with the value NN, the number of elements in the pattern. Within this loop, characters from the keyboard are accepted or rejected. The GET function returns with a character from the keyboard. If no key has been pressed, then the string T$ is assigned a null value. As long as T$ continues to be a null string, the program will keep looping on line 2060. As soon as a key is pressed, the program continues at line 2070. Normally, when the GET function is used, the cursor does not flash. POKEing a 0 into CF (a constant set to 204 in the initialization) starts the cursor flashing; POKEing a 1 turns it off. It must be turned off between GETs to avoid depositing cursor characters in unwanted places.

Two special characters '←' and '?' are tested. On '←', the loop is terminated by setting I to NN and executing a NEXT statement. The GOTO 2030 starts the loop over again. If we had failed to terminate the loop (by omitting the I = NN and NEXT statements) the user would be able to crash the program by repeatedly hitting the '←' key. BASIC keeps track of each FOR...NEXT loop in an area of memory called the stack. If we don't terminate a loop, that information continues to occupy space on the stack. Repeated calls to 2030 with the '←' key will continue to build up new FOR...NEXT information on the stack until there is no room left. At this point the program crashes with an ?OUT OF MEMORY ERROR. The '?' character is dealt with similarly. The FOR...NEXT loop is terminated and a RETURN is made to the program mainline.

Other characters are converted in line 2100 to their numeric codes with the ASC function. The code for the letter A is 65, so subtracting 64 converts letters into numbers beginning with 1. If T is less than 1 or greater than the number of letters allowed in the pattern, then BK$ (a constant defined as two [CL] characters) is printed and the program branches to 2050 to GET another character. If the character is accepted, then the appropriate colored letter block OB$(T) is printed and the number T is stored in the current element of the guess pattern GU(I). RETURN takes the flow back to the mainline.

Matching Routines (3000-3580): These routines are described in more detail in the main text under the section "The Matching Process."

3010-3030 clear the flag arrays PG( ) and PF( ) by setting them to zero. 3050-3070 advance, position by position, through the secret pattern R( ) and guess pattern GU( ) arrays checking for matches. If a match is found, the position match counter PM is incremented and the corresponding flags are set to – 1.

Line 3500-3580 check all the other possibilities for matches. The flags are used to cross off elements as they are matched. Some economy is achieved by skipping over crossed-off elements (lines 3520 and 3540) and by terminating the inside loop as soon as a match is found (J = NN at the end of line 3550).

Input Pattern with Two Players (4000-4160): After the instructions are displayed, this routine accepts letters one-by-one until the pattern is filled. It is similar to the guess-processing routine (2000-2130). Instead of filling the guess array, the secret pattern array R( ) is filled. See the description above for details.

ANY KEY WHEN READY (5000-5020): The string CN$ is a constant defined in the initialization routine. The result is to print the

message 'ANY KEY WHEN READY' at the bottom of the screen. The one-line GET loop must receive a key before a RETURN.

Congratulation Routine (6000-6190): 6010-6060 use the number of guesses GN to determine the congratulation message. The array G( ) is set up for each version of the game in 7100, 7200, or 7300. The messages MS$( ) are set up as constant strings during initialization. By comparing GN to each cutoff value G( ) with the < = (less than or equal to) operation, the subscript MS for the array MS$( ) is determined.

Lines 6070-6180 display the congratulation screen, alternating the message and the number of guesses between reversed and normal characters. The use of I as a flag is discussed above under "Flags." If I = −1 then the [RVS] character is printed. Its value alternates between −1 and 0.

The string functions STR$ and MID$ applied to GN in line 6130 make the display of the number of guesses more attractive. The STR$ function converts the number GN to a string of characters. Positive numbers leave a space in front of the numeric characters that normally would be occupied by the '−' character. To get just the numeric part of the string the MID$ is used in a special way to get all the characters from the second position on. Normally the items included in the parentheses after MID$ are the name of the string, the character position to start, and the number of characters to extract. If just the first two items are included, then the remainder of the string is the result. Specifying 2 for the second parameter converts the string of the number of guesses to the same string without the leading space.

Line 6140 is a FOR...NEXT loop that does nothing between the FOR and the NEXT! By adjusting the number after the TO, you can achieve a delay in the program of nearly any desired time. Here it controls the rate of the flashing.

The GET function is used in line 6170 in a way opposite to its use in 2050, 4090, 5000, and 7060. As long as no key is pressed (T$ does not equal " "), the message continues to flash. When a key is pressed, the RETURN instruction is executed.

Select Game and Display Instructions (7000-7440): 7010-7040 display a menu listing the different games available. Line 7060 awaits a key, which is converted to a number and tested against the range of the menu in line 7070. If the key is out of range, then the program branches back to 7060 for another key. The ON...GOSUB instruction in line 7080 calls 7100 if T is 1, 7200 if T is 2, or 7300 if T is 3.

Each of these set-up-and-display routines establishes N (the number of letters allowed in the game), RP (the flag determining whether or not repeats are allowed), and G( ) (the array of guess number cutoff values for the congratulation messages). Then the name of the game is displayed. Next subroutine 7400, which displays parts of the instructions common to all games, is called. Finally, the rule regarding repeats is printed in the proper place.

Subroutine 7400 first prints the colored letter blocks corresponding to the number of letters allowed (N). If the number allowed is four, the first four letter blocks are printed. Then the portion of the directions common to all versions of the game is printed.

Get Number of Players (7500-7520): This subroutine is called once when the program is first run. It uses subroutine 5010 (5000 without the ANY KEY WHEN READY message) to GET a key. Only 1 or 2 is accepted and the value is returned in NP.

Initialization (8000-8080): Sets up constants used in the program. See variable usage table for descriptions of the variables.

Print Pattern on Give-up (9000-9040): The secret pattern is printed out in the appropriate colored letter blocks, using the secret pattern array R( ). Subroutine 5000 is used to wait for a key before starting a new game.

Running MASTER on other Commodore Computers

The program will run as it is on a Commodore 64. For PET Computers, change the value of CF from 204 to 167 (line 8040). Also, omit the color control codes.

MASTER Listing

```
10 GOSUB 8000:GOSUB7500:REM SET UP CONSTANTS,
                            SELECT # OF PLAYERS
100 GOSUB7000:GOSUB5000:REM SELECT GAME LEVEL,
                           WAIT FOR KEY
110  ON NP
      GOSUB 1000,4000:
        REM  RANDOM OR PLAYER-INPUT PATTERN
120 GN=1
130 GOSUB2000:GOTO150:REM HEADER, ENTER GUESS
140 GOSUB2020:REM ENTER GUESS (NO HEADER)
150  IF T$="?"
      THEN GOSUB 9000:
        GOTO 100
160 PM=0:OM=0
170  GOSUB 3000:
      REM  PROCESS GUESS, RETURN PM,OM
180  IF PM=NN
      THEN GOSUB 6000:
        GOTO 100
190  PRINT TAB(12)"[BLK] "PM;OM
200  GN=GN+1:
      GOTO 140

1000 REM  GENERATE RANDOM NUMBERS
1010 FOR I=1 TO NN
1020   RN=INT(RND(1)*N+1)
```

MASTER Listing *(continued)*

```
1030   IF RP
       THEN 1090
1040   RQ=0:
       IF I=1
       THEN 1090
1050   FOR J=1 TO I-1
1060     IF RN=R(J)
         THEN RQ=-1:
           J=I-1
1070   NEXT J
1080   IF RQ
       THEN 1020
1090   R(I)=RN
1100 NEXT I
1110 RETURN
2000 REM  PRINT HEADER
2010 PRINT "[CLR][RVS][PUR]SELECT LETTER
     [OFF] ON OFF[CD]"
2020 REM  PROCESS GUESS
2030 FOR I=1 TO NN
2040   PRINT FD$;
2050   POKE CF,0
2060   GET T$:
       IF T$=""
       THEN 2060
2070   POKE CF,1
2080   IF T$="←"
       THEN PRINT CR$"[CU]              [CU]";:
         I=NN:
       NEXT :
       GOTO 2030
2090 IF T$="?"
     THEN I=NN:
       NEXT :
       GOTO 2140
2100 T=ASC(T$)-64:
     IF T<1 OR T>N
     THEN PRINT BK$;:
       GOTO 2040
2110 PRINT OB$(T);
2120 GU(I)=T
2130 NEXT I
2140 RETURN


3000 REM  CLEAR FLAGS
3010 FOR I=1 TO NN
3020   PF(I)=0:
       PG(I)=0
3030 NEXT I
3040 REM  CHECK FOR POSITION MATCHES
3050 FOR I=1 TO NN
3060   IF R(I)=GU(I)
       THEN PF(I)=-1:
         PG(I)=-1:
         PM=PM+1
3070 NEXT I
3500 REM  CHECK FOR OTHER MATCHES
3510 FOR I=1 TO NN
3520   IF PG(I)
       THEN 3570
3530   FOR J=1 TO NN
3540     IF PF(J)
         THEN 3560
3550     IF R(I)=GU(J)
         THEN OM=OM+1:
           PF(J)=-1:
           PG(I)=-1:
           J=NN
3560   NEXT J
3570 NEXT I
3580 RETURN
4000 REM  INPUT PATTERN
4010 PRINT "[CLR]ONE PLAYER ENTERS"
4020 PRINT " PATTERN"
4030 PRINT "WHILE OTHER PLAYER"
4040 PRINT " LOOKS AWAY"
4050 PRINT "[CD]ENTER PATTERN:"
4060 FOR I=1 TO NN
```

MASTER Listing *(continued)*

```
4070   PRINT FD$;
4080   POKE CF,0
4090   GET T$:
       IF T$=""
       THEN 4090
4100   POKE CF,1
4110   IF T$="←"
       THEN PRINT BK$:
         I=NN:
       NEXT :
       GOTO 4060
4120 T=ASC(T$)-64:
     IF T<1 OR T>N
     THEN 4080
4130 PRINT OB$(T);
4140 R(I)=T
4150 NEXT
4160 RETURN
5000 PRINT CN$
5010 GET T$:
     IF T$=""
     THEN 5010
5020 RETURN

6000 REM  CONGRATULATIONS
6010 IF GN=1
     THEN MS=1:
       GOTO 6070
6020 IF GN<=G(1)
     THEN MS=2:
       GOTO 6070
6030 IF GN<=G(2)
     THEN MS=3:
       GOTO 6070
6040 IF GN<=G(3)
     THEN MS=4:
       GOTO 6070
6050 IF GN<=G(4)
     THEN MS=5:
       GOTO 6070
6060 MS=6
6070 PRINT "[CLR]";
6080 FOR I=-1 TO 0
6090   IF I
       THEN PRINT "[RVS]";
6100   PRINT "[HOME][RED]"MS$(MS)
6110   PRINT "[CD][PUR]YOU TOOK [CYN]";
6120   IF I
       THEN PRINT "[RVS]";
6130   PRINT MID$(STR$(GN),2);"[OFF]
            [PUR] TRIES!"
6140   FOR J=1 TO 200:
       NEXT J
6150 NEXT I
6160 PRINT CN$
6170 GET T$:
     IF T$<>""
     THEN RETURN
6180 GOTO 6080
7000 REM  PROCESS INITIAL CONDITIONS
7010 PRINT "[CLR]SELECT GAME:"
7020 PRINT "[CD][CR][CR][RVS]1[OFF] EASY"
7030 PRINT "[CD][CR][CR][RVS]2[OFF] MID"
7040 PRINT "[CD][CR][CR][RVS]3[OFF] HARD"
7060 GET T$:
     IF T$=""
     THEN 7060
7070 T=VAL(T$):
     IF T<1 OR T>3
     THEN 7060
7080 ON T
     GOSUB 7100,7200,7300
7090 RETURN
7100 N=4:RP=0:G(1)=3:G(2)=5:G(3)=7:
     G(4)=10:G(5)=15
7110 PRINT "[CLR]EASY GAME:"
7120 GOSUB 7400
```

MASTER Listing *(continued)*

```
7130 PRINT "[CD][CR][CR]ONLY ONCE"
7140 RETURN
7200 N=4:RP=-1:G(1)=4:G(2)=6:G(3)=8:
      G(4)=12:G(5)=18
7210 PRINT "[CLR]MID GAME:"
7220 GOSUB 7400
7230 PRINT "[CD][CR][CR]MORE THAN ONCE"
7240 RETURN
7300 N=6:RP=-1:G(1)=5:G(2)=7:G(3)=10:
      G(4)=15:G(5)=20
7310 PRINT "[CLR]HARD GAME:"
7320 GOSUB 7400
7330 PRINT "[CD][CR][CR]MORE THAN ONCE"
7340 RETURN
7400 PRINT "[CD][CR][CR]";:
     FOR I=1 TO N:
       PRINT OB$(I)" ";:
     NEXT :
     PRINT " ALLOWED"
7410 PRINT "[CD][CR][CR]EACH MAY  BE USED"
7420 PRINT "[6 CD][RVS]←[OFF] TO CLEAR GUESS"
7430 PRINT "[CD][RVS]?[OFF] TO GIVE UP[HOME]
      [CD][CD][CD][CD]"
7440 RETURN
7500 PRINT "[CLR][RVS]1[OFF] OR [RVS]2
     [OFF] PLAYERS?"
7510 GOSUB 5010:
     IF T$<"1" OR T$>"2"
     THEN 7510
7520 NP=VAL(T$):
     RETURN
```

MASTER Listing *(continued)*

```
8000 REM  SET-UP OF CONSTANTS
8010 OB$(1)="[RVS][BLK]A[OFF]":
     OB$(2)="[RVS][RED]B[OFF]":
     OB$(3)="[RVS][CYN]C[OFF]"
8020 OB$(4)="[RVS][PUR]D[OFF]":
     OB$(5)="[RVS][GRN]E[OFF]":
     OB$(6)="[RVS][BLU]F[OFF]"
8030 CN$="[HOME][20CD][CR][CR][RVS]ANY KEY
     WHEN READY"
8040 NN=4:
     FD$="[BLK] >":
     BK$="[CL][CL]":
     CR$=CHR$(13):
     CF=204:
     I=RND(-TI)
8060 MS$(1)="A PSYCHIC!":
     MS$(2)="EXCELLENT!":
     MS$(3)="VERY GOOD!"
8070 MS$(4)="GOOD":
     MS$(5)="FAIR":
     MS$(6)="TRY,TRY,TRY AGAIN!"
8080 RETURN

9000 REM  PRINT PATTERN ON GIVE-UP
9010 PRINT CR$"[CU][RVS]GIVE UP?
     [OFF] PATTERN IS:"
9020 FOR I=1 TO NN:
       PRINT " >"OB$(R(I))"[OFF]";:
     NEXT
9030 GOSUB 5000
9040 RETURN
```

MASTER for the APPLE

```
10  GOSUB 8ØØØ: GOSUB 75ØØ
1ØØ  GOSUB 7ØØØ: GOSUB 5ØØØ: ON N
     P GOSUB 1ØØØ,4ØØØ:GN = 1: GOSUB
     2ØØØ: GOTO 15Ø
14Ø  GOSUB 2Ø2Ø
15Ø  IF T$ = "?" THEN  GOSUB 9ØØØ
     : GOTO 1ØØ
16Ø PM = Ø:OM = Ø: GOSUB 3ØØØ: IF
     PM = NN THEN  GOSUB 6ØØØ: GOTO 1ØØ
19Ø  HTAB 22: PRINT PM"     "OM"
        "NN - (PM + OM):GN = GN +
     1: GOTO 14Ø
1ØØØ  FOR I = 1 TO NN
1Ø2Ø RN =  INT ( RND (1) * N + 1)
     : IF RP THEN 1Ø9Ø
1Ø4Ø RQ = Ø: FOR J = 1 TO I: IF R
     N = R(J) THEN RQ = 1
1Ø7Ø  NEXT : IF RQ THEN 1Ø2Ø
1Ø9Ø R(I) = RN: NEXT : RETURN
2ØØØ  HOME : PRINT "SELECT LETTER
          ON    OFF  WRONG"
2Ø2Ø  PRINT : POKE 34,1: FOR I =
     1 TO NN
2Ø4Ø  PRINT ">";: GET T$: IF T$ =
      CHR$ (8) THEN  HTAB 1: CALL
      - 868:I = 1: GOTO 2Ø4Ø
2Ø8Ø  IF T$ = "?" THEN I = NN: GOTO
     212Ø
2Ø9Ø T =  ASC (T$) - 64: IF T < 1
      OR T > N THEN  PRINT  CHR$
     (8);: GOTO 2Ø4Ø
21ØØ  INVERSE : PRINT T$;: NORMAL
     : PRINT " ";:GU(I) = T
212Ø  NEXT : RETURN
3ØØØ  FOR I = 1 TO NN:PF(I) = Ø:P
     G(I) = Ø: NEXT : FOR I = 1 TO
     NN: IF R(I) = GU(I) THEN PF(
     I) = 1:PG(I) = 1:PM = PM + 1
3Ø6Ø  NEXT : FOR I = 1 TO NN: IF
     PG(I) THEN 357Ø
353Ø  FOR J = 1 TO NN: IF PF(J) THEN
     356Ø
355Ø  IF R(I) = GU(J) THEN OM = O
     M + 1:PF(J) = 1:PG(I) = 1:J = NN
356Ø  NEXT
357Ø  NEXT : RETURN
4ØØØ  HOME : PRINT "ONE PLAYER EN
     TERS PATTERN": PRINT "WHILE
     OTHER PLAYER LOOKS AWAY.": PRINT
     : PRINT "ENTER PATTERN:": FOR
     I = 1 TO NN
4Ø7Ø  PRINT ">";
4Ø9Ø  GET T$: IF T$ =  CHR$ (8) THEN
      HTAB 1: CALL  - 868:I = 1: GOTO 4Ø7Ø
412Ø T =  ASC (T$) - 64: IF T < 1
      OR T > N THEN 4Ø9Ø
413Ø  PRINT  CHR$ (95);:R(I) = T:
      NEXT : RETURN
5ØØØ  VTAB 23: HTAB 1Ø: FLASH : PRINT
     " ANY KEY WHEN READY";: GET
     T$: NORMAL : RETURN
6ØØØ  TEXT : HOME : VTAB 5: FLASH
     : FOR I = 1 TO 6: IF GN < G(
     I) THEN MS = I:I = 6
6Ø3Ø  NEXT : PRINT MS$(MS)" ";: NORMAL
     : PRINT "YOU TOOK "GN" TRIES
     !": GOSUB 5ØØØ: RETURN
7ØØØ  HOME : VTAB 5: PRINT "SELEC
     T GAME:": PRINT : PRINT : INVERSE
     : PRINT "1";: NORMAL : PRINT
     " EASY": PRINT : INVERSE : PRINT
     "2";: NORMAL : PRINT " MIDDL
     E": PRINT : INVERSE : PRINT
     "3";: NORMAL : PRINT " HARD"
     : PRINT :G(6) = 5ØØØØ
7Ø6Ø  PRINT "WHICH?";: GET T$:T =
      VAL (T$): IF T < 1 OR T > 3
      THEN 7Ø6Ø
7Ø8Ø  ON T GOSUB 71ØØ,72ØØ,73ØØ: RETURN
71ØØ N = 4:RP = Ø:G(1) = 2:G(2) =
     4:G(3) = 6:G(4) = 8:G(5) = 1
     1: HOME : PRINT "EASY GAME:"
     : GOSUB 74ØØ: VTAB 5: HTAB 1
     8: PRINT "ONLY ONCE": RETURN
72ØØ N = 4:RP = 1:G(1) = 2:G(2) =
     5:G(3) = 7:G(4) = 9:G(5) = 1
     3: HOME : PRINT "MIDDLE GAME
     :": GOSUB 74ØØ: VTAB 5: HTAB
     18: PRINT "MORE THAN ONCE": RETURN
73ØØ N = 6:RP = 1:G(1) = 2:G(2) =
     6:G(3) = 8:G(4) = 11:G(5) =
     16: HOME : PRINT "HARD GAME:
     ": GOSUB 74ØØ: VTAB 5: HTAB
     18: PRINT "MORE THAN ONCE": RETURN
74ØØ  PRINT : PRINT "  ";: FOR I =
     1 TO N: INVERSE : PRINT  CHR$
     (64 + I);: NORMAL : PRINT "
      ";: NEXT : PRINT "ALLOWED":
      PRINT : PRINT "EACH MAY 8E
     USED": VTAB 15: PRINT "  <-
     TO CLEAR GUESS": PRINT : PRINT
     "  ?  TO GIVE UP": RETURN
75ØØ  HOME : PRINT : PRINT "  ";:
      INVERSE : PRINT "1";: NORMAL
     : PRINT " OR ";: INVERSE : PRINT
     "2";: NORMAL : PRINT " PLAYERS?"
751Ø  PRINT "  WHICH?";: GET T$:N
     P =  VAL (T$): IF NP < 1 OR
     NP > 2 THEN 751Ø
753Ø  RETURN
8ØØØ NN = 4:MS$(1) = "A PSYCHIC":
     MS$(2) = "EXCELLENT!":MS$(3)
      = "VERY GOOD!":MS$(4) = "GO
     OD":MS$(5) = "FAIR":MS$(6) =
     "TRY, TRY, TRY AGAIN": RETURN
9ØØØ  TEXT : HOME : PRINT "GIVE U
     P?": PRINT "PATTERN IS:": FOR
     I = 1 TO NN: PRINT ">";: INVERSE
     : PRINT  CHR$ (64 + R(I));: NORMAL :
  PRINT " ";: NEXT : GOSUB 5ØØØ: RETURN
```

Circle No. 46

LEARNING CENTER

Conservation of Momentum for ATARI and COMMODORE

by Jerry Faughn

Conservation laws, such as the conservation of momentum, are among the most important concepts covered in an introductory level physics course. This program examines the conservation of momentum as applied to collision problems. Two cars are sent toward each other to collide under a variety of conditions selected by the viewer. Programs such as this can be a valuable instructional tool used in a physics class, either as a demonstration or as an interactive tutorial program for a student. But, you don't have to be a physics student to have fun playing around with it.

One of the parameters that the viewer must choose is a value for the coefficient of restitution. This number can range between the extremes of zero and one. If the coefficient of restitution is selected to be one, the collision is said to be perfectly elastic. That is, when the objects collide there is no distortion or bending of the objects. Such conditions obviously do not prevail in the real world of collisions between cars, but they can and do occur in collisions between atoms and subatomic atomic particles.

In the real world, collisions between very rigid objects, such as billiard balls, are highly elastic. At the other extreme are collisions for which the coefficient of restitution is zero; these are called perfectly inelastic collisions. Such collisions are characterized by the two objects sticking together and moving as a unit after the collision. This program can handle elastic and perfectly inelastic collisions as well as the broad spectrum between these two extremes.

Two typical trial situations that you might want to examine use the following parameters. Trial one: coefficient of restitution = 1, mass of blue car = 20, mass of orange car = 4. Trial two: coefficient of restitution = 0, mass of blue car = 10.

This program used player-missile graphics and is explained via remarks within the program. *Editor's note: The Commodore 64 version uses the C64 Sprite graphics. The two movable-object-block graphics systems have a number of similarities, as well as differences.*

```
9 REM PRINT TITLE
10 GRAPHICS 18:SETCOLOR 4,2,2
20 POSITION 4,4:PRINT #6;"CONSERVATION"
30 POSITION 9,5:? #6;"OF"
35 POSITION 6,6:? #6;"MOMENTUM"
40 FOR N=1 TO 1500:NEXT N
49 REM SET UP EXAMPLE COLLISION
50 GRAPHICS 0:SETCOLOR 4,14,4:SETCOLOR 2,8,4
60 POKE 752,1:? "}"
70 ? ;"HERE IS AN EXAMPLE OF A COLLISION"
75 ? ;"LIKE YOU WILL SEE"
80 ? ;"THE BLUE CAR AND THE ORANGE CAR"
85 ? ;"HAVE EQUAL MASSES. BLUE HAS A SPEED"
86 ? ;"OF 2 M/S.  ORANGE 4 M/S."
90 ? ;"COEFFICIENT OF RESTITUTION IS ONE.":FOR N=1 TO 1500:NEXT N
99 REM PARAMETERS FOR INITIAL COLLISION
100 COR=1:M1=50:M2=50:V1I=2:V2I=4
110 GO TO 500
200 GRAPHICS 16:SETCOLOR 4,14,4:SETCOLOR 2,8,4:REM INPUT PARAMETERS FOR NEXT COL
LISION
204 POKE 752,1:? "}"
207 TRAP 215
210 ? "WHAT COEFFICIENT OF RESTITUTION":? "DO YOU WANT?":INPUT COR
211 IF COR<0 THEN GO TO 1000
212 IF COR>1 THEN GO TO 1000
213 GO TO 216
215 ? "INPUT A NUMBER 0 THRU 1":GO TO 210
216 TRAP 1050
217 ? "WHAT WILL BE THE MASS OF THE BLUE CAR? ":INPUT M1
218 IF M1<=0 OR M1>50 THEN GO TO 1050
225 TRAP 1060
227 ? "WHAT WILL BE THE MASS OF":? "THE ORANGE CAR?":INPUT M2
228 IF M2<=0 OR M2>50 THEN GO TO 1060
240 TRAP 1075
245 ? "INPUT VELOCITY OF BLUE CAR."
246 ? "USE POSITIVE NUMBERS BETWEEN 0 AND 10."
247 INPUT V1I
248 IF V1I<0 THEN GO TO 1075
249 IF V1I>10 THEN GO TO 1075
250 TRAP 1100
252 ? "INPUT VELOCITY OF ORANGE CAR."
253 ? "USE POSITIVE NUMBERS BETWEEN 0 AND 10.":INPUT V2I
254 IF V2I<0 THEN GO TO 1100
255 IF V2I>10 THEN GO TO 1100
260 HPO1=135-(V1I/(V1I+V2I))*85
265 HPO2=135+(V2I/(V2I+V1I))*85
270 GO TO 501
500 HPO1=50:HPO2=220:REM INITIAL HOR POS OF CARS
501 POKE 53248,HPO1:POKE 53249,HPO2
509 REM CHOOSE REGULAR PLAYFIELD AND COLOR OF CARS
510 GRAPHICS 23:SETCOLOR 4,10,4:POKE 559,62:POKE 704,116:POKE 705,40
515 SETCOLOR 0,0,4:COLOR 1:FOR Z=42 TO 50:PLOT 0,Z:DRAWTO 159,Z:NEXT Z:REM DRAW
HIGHWAY
520 I=PEEK(106)-32:REM RESERVE SPACE FOR P/M GRAPHICS
530 POKE 54279,I:REM PLACE ADR IN P/M BASE ADDRESS REGISTER
540 POKE 53278,0:REM SET COLLISION REGISTER TO ZERO
550 POKE 53277,3:REM ENABLE P/M GRAPHICS
559 REM IF CAR 1 IS MUCH MORE MASSIVE THAN CAR 2, CAR 1 IS TWICE NORMAL SIZE
560 IF M1/M2>5 THEN POKE 53256,1:POKE 53257,0:GO TO 600
570 IF M2/M1>5 THEN POKE 53256,0:POKE 53257,1:GO TO 600:REM SEE NOTE ON STATEMEN
T 560
580 POKE 53256,0:POKE 53257,0
600 J=I*256+1024:REM LOCATION OF PLAYER 0
610 FOR Y=J+120 TO J+127:REM READ IN SHAPE OF CAR1
620 READ Z:POKE Y,Z:NEXT Y
630 DATA 0,255,125,223,223,223,125,255,0,255,125,251,251,251,125,255,0
640 J=I*256+1280:REM MEM LOCATION OF PLAYER 1
650 FOR Y=J+120 TO J+127:READ Z:REM READ IN SHAPE OF CAR 2
651 POKE Y,Z:NEXT Y
652 RESTORE
660 FOR X=1 TO 220:REM MOVE CARS TOWARD EACH OTHER
670 PO1=HPO1+V1I*X/5
680 PO2=HPO2-V2I*X/5
690 POKE 53248,PO1:POKE 53249,PO2
700 IF PEEK(53260)<>0 THEN GO TO 720:REM CHECK FOR COLLISION
710 NEXT X
720 V2F=M1*(COR*V1I-COR*(-V2I)+V1I)+M2*(-V2I):REM FIND VELOCITY OF EACH CAR AFTE
R COLLISION
730 M=M1+M2:POKE (53260),0
740 V2F=V2F/M
750 V1F=V2F-COR*V1I-COR*V2I
760 FOR X=1 TO 500:REM MOVE CARS AFTER COLLISION
770 POKE 53249,PO2+V2F*X/5
780 POKE 53248,PO1+V1F*X/5
790 IF PO2+V2F*X<20 THEN GO TO 900:REM STOP MOVEMENT
800 IF PO2+V2F*X>245 THEN GO TO 900
810 IF PO1+V1F*X<20 THEN GO TO 900
820 IF PO1+V1F*X>270 THEN GO TO 900
830 NEXT X
900 POKE 53277,1:REM TURN OFF P/M GRAPHICS
910 GRAPHICS 16:SETCOLOR 4,14,4:SETCOLOR 2,8,4:REM SET UP SCREEN FOR DISPLAY OF
VELOCITIES                    0
920 POKE 752,1:? "}"
930 ? "FINAL VELOCITY OF BLUE CAR  IS ";(INT(V1F*100))/100
940 ? "FINAL VELOCITY OF ORANGE CAR  IS ";(INT(V2F*100))/100
950 ? "IF YOU WOULD LIKE TO TRY":? "ANOTHER COLLISION PRESS SPACE BAR"
960 IF PEEK(764)<>33 THEN GO TO 960
970 GO TO 200
1000 ? "COEFFICIENT MUST BE BETWEEN ZERO AND ONE.":FOR N=1 TO 50:NEXT N:GO TO 21
0
1050 ? "MASS MUST BE A POSITIVE NUMBER"
1051 ? "BETWEEN 1 AND 50":FOR N=1 TO 50:NEXT N:GO TO 217
1060 ? "MASS MUST BE A POSITIVE NUMBER"
1061 ? "BETWEEN 1 AND 50":FOR N=1 TO 50:NEXT N:GO TO 227
1075 ? "VELOCITY MUST BE A POSITIVE NUMBER."
1076 ? "BETWEEN 0 AND 10":FOR N=1 TO 50:NEXT N:GO TO 245
1100 ? "VELOCITY MUST BE A POSITIVE NUMBER "
1101 ? "BETWEEN 0 AND 10":FOR N=1 TO 50:NEXT N:GO TO 252
```

```
1 REM CONSERVATION OF MOMENTUM
2 REM BY JERRY FAUGHN
3 REM C64 VERSION BY LOREN WRIGHT
10 GOSUB4000
15 PRINT""TAB(14)"CONSERVATION"
20 PRINT""TAB(19)"OF"
25 PRINT""TAB(16)"MOMENTUM"
30 FOR1=1TO2000:NEXT
35 PRINT"HERE IS A SAMPLE COLLISION."
40 PRINT"THE MASSES OF THE TWO CARS"
45 PRINT" ARE EQUAL."
50 PRINT"THE SPEED OF THE RED CAR IS"
55 PRINT" TWICE THAT OF THE BLUE CAR"
60 FOR1=1TO2000:NEXT
65 CR=1:M1=50:M2=50:V1=2:V2=4
70 GOTO210
100 REM PROGRAM MAIN LINE
110 PRINT"WHAT COEFFICIENT OF RESTITUTION?":INPUT" (VALUE 0 TO 1)";CR
120 IF(CR<0)OR(CR>1)THEN110
130 PRINT"WHAT IS THE MASS OF THE BLUE CAR?":INPUT" (VALUE 1 TO 50)";M1
140 IF(M1<1)OR(M1>50)THEN130
150 PRINT"WHAT IS THE MASS OF THE RED CAR?":INPUT" (VALUE 1 TO 50)";M2
160 IF(M2<1)OR(M2>50)THEN150
170 INPUT"VELOCITY OF BLUE CAR (0 TO 10)";V1
180 IF(V1<0)OR(V1>10)THEN170
190 INPUT"VELOCITY OF RED CAR (0 TO 10)";V2
200 IF(V2<0)OR(V2>10)THEN190
205 REM SET STARTING POSITIONS SO THAT
206 REM  COLLISION OCCURS AT SCREEN CENTER
210 H1=174-(V1/(V1+V2))*150
220 H2=174+(V2/(V1+V2))*150
230 GOSUB3000:GOSUB1000
235 GOSUB2000:POKEV+21,0
240 PRINT"FINAL VELOCITY OF BLUE CAR IS ";(INT(VA*100+.5))/100
250 PRINT"FINAL VELOCITY OF RED CAR IS ";(INT(VB*100+.5))/100
260 GOSUB2000
270 GOTO100
1000 REM PERFORM COLLISION
1010 REM EXPAND EITHER CAR IF 5X HEAVIER
1020 Z1=0:IFM1/M2>5THENZ1=1
1030 Z2=0:IFM2/M1>5THENZ2=2
1040 POKEV+29,Z1+Z2
1050 POKEV+30,0
1100 REM ADVANCE CARS UNTIL COLLISION
1110 X=0
1120 P1=H1+V1*X/5
1130 P2=H2-V2*X/5
1140 Q1=0:IFP1>255THENP1=P1-255:Q1=1
1150 Q2=0:IFP2>255THENP2=P2-255:Q2=2
1160 POKEV,P1:POKEV+2,P2:POKEV+16,Q1+Q2
1170 IFPEEK(V+30)THENPOKEV+30,0:GOTO1210
1180 X=X+1
1190 GOTO1120
1200 REM FIND FINAL VELOCITIES
1210 VB=M1*(CR*V1-CR*(-V2)+V1)+M2*(-V2)
1220 VB=VB/(M1+M2)
1230 VA=VB-CR*V1-CR*V2
1300 REM MOVE CARS UNTIL ONE REACHES EDGE
1310 X=0
1320 PB=P2+VB*X/5
1330 PA=P1+VA*X/5
1340 PD=PB:Q2=0:IFPB>255THENPD=PD-255:Q2=2
1350 PC=PA:Q1=0:IFPA>255THENPC=PC-255:Q1=1
1360 POKEV,PC:POKEV+2,PD:POKEV+16,Q1+Q2
1370 IF(PA<24)OR(PA>323)THEN4500
1380 IF(PB<24)OR(PB>323)THEN4500
1390 GETT$:IFT$<>""THEN1500
1400 X=X+1
1410 GOTO1320
1500 RETURN
2000 PRINT""TAB(10)"ANY KEY TO CONTINUE"
2010 GETT$:IFT$=""THEN2010
2020 RETURN
3000 REM SET UP SCREEN
3010 PRINT"":POKEV+33,0
3020 PRINT""BL$BL$
3030 POKEV,H1:POKEV+2,H2AND255:POKEV+16,-((H2>255)*2+(H1>255))
3040 POKEV+21,3
3050 RETURN
4000 REM SET UP CONSTANTS & SPRITES
4010 V=53248:BL$=" | | | | | | | | | | | | | | | | | | | |"
4020 B=192*64
4030 REM CLEAR SPACE FOR TWO SPRITES
4040 FOR1=0TO127:POKEB+I,0:NEXT
4050 REM READ PATTERNS FROM DATA
4060 REM  AND STORE IN MEMORY
4070 FORI=0TO26:READA:POKEB+I,A:NEXT
4080 B=193*64
4090 FORI=0TO26:READA:POKEB+I,A:NEXT
4100 REM SET UP POINTERS, COLORS, & Y POS
4110 POKEV+39,6:POKEV+40,2
4120 POKE2040,192:POKE2041,193
4130 POKEV+1,149:POKEV+3,149
4140 POKEV+21,0
4150 POKEV+33,0:POKEV+32,1
4160 RETURN
5000 DATA62,31,0,28,14,0,127,255,128,255,231,240,255,249,240,255
5010 DATA231,240,127,255,128,28,14,0,62,31,0
5020 DATA15,135,192,7,3,128,31,255,224,254,127,240,249,255,240,254
5030 DATA127,240,31,255,224,7,3,128,15,135,192
```

MICRO™

Is a Number a Number?

by Phil Daley

On your toes now. Here is a quick quiz: How much is 7 + 5? a) 12, b) 14, c) C, d) all of the above, e) I don't know. If you answered d, then you may skip the rest of this article, unless you made a lucky guess.

The answer depends upon the base of the number system you are working in. Normally, when you are working with everyday decimal numbers, you are using the base of 10. That means that each place to the left (or right) of the decimal point represents a power of ten. The first place to the left of the decimal point represents how many 10^0's there are in that number. For instance, a '7' indicates 7 * (10^0). Since 10^0 = 1, 7 * 1 = 7. The number 7 = 7! When working with base 10 numbers things seem pretty easy, but humor me and follow along; it will get tougher.

What about 17? The 1 represents 1 * 10^1, or 10. Add the 7 and you get 17. Each place farther to the left of the decimal point increases the power of 10 — 10^2, 10^3, 10^4 ... etc. This gives you the one's place, ten's place, hundred's place, and so on.

What happens when you use a base that is different than 10 — for instance 16? I choose 16 as an example since it is the basis of the hexadecimal system, which computer people use all the time as it is a more convenient system with which to work. Now the first place to the left of the point represents 16^0, or ones. Sounds familiar, right? However, how many numbers can be counted until you have to carry over to the next place? In decimal you count to 9 and then carry one to the ten's column. In hexadecimal you count to 15 before the carry to the next column. This is going to cause trouble. What happens after 9? You use letters! The first six letters of the alphabet represent the numbers 10-15. Counting in hexadecimal goes 1, 2, ... 8, 9, A, B, C, D, E, F, 10, 11 etc.

The second place to the left counts as 16^1, or 16's, the third place (16^2), 256's, and the fourth place (16^3), 4096's. This is normally as high as you need to go on microcomputers.

How much is 7 + 5 (in base 16)? Now you see how answer C derives. Standard convention uses the $ sign to indicate hexadecimal numbers. What does $5CE equal? The answer is 5 * 256 + 12 * 16 + 14 * 1. 1486 in base 10.

A computer doesn't really understand hexadecimal. A hardware circuit in the computer has only two states — on and off or high and low. These are represented by 1's and 0's. Since you can count only to one before you have to carry to the next place, this is working with base 2 numbers. Binary numbers are the code that microcomputers understand; unfortunately they are not recognized easily by humans and are, therefore, prone to error in reading and typing. It is simple to transpose two digits in a number like %1110010101100110. The % sign is standard to indicate a binary number. So that number is equivalent to 58726 in base 10. (1 * 32768 + 1 * 16384 + 1 * 8192 + 0 * 4096 + 0 * 2048 + 1 * 1024 + 0 * 512 + 1 * 256 + 0 * 128 + 1 * 64 + 1 * 32 + 0 * 16 + 0 * 8 + 1 * 4 + 1 * 2 + 0 * 1). Not the easiest conversion.

A number like %1000000000000000 is equivalent to 32768 in base 10, which is not too memorable. In hexadecimal, it's equivalent to $8000. Now perhaps you can see why computer people use hexadecimal. When talking about a microcomputer's memory map, pages of memory are used as a convenient way to locate various usages. For instance, in 6502 computers page 0 is used by the system for pointer storage (due to zero page addressing), page 1 is used for the system stack, and page 2 is sometimes used for the input buffer. In decimal, that would convert to page 0 = 0 to 255, page 1 = 256 to 511, and page 2 = 512 to 767. Hexadecimal notation is much easier to remember — page 0 = $0 to $FF, page 1 = $100 to $1FF, and page 2 = $200 to $2FF.

Even if you can't see much use for the different numbering systems now, when you start to work with machine language you may wonder why anyone works with base 10. This program converts any base (2, 8, 10, 16) into all the others. The routines do the conversions the same way you would do them by hand; you can learn the conversion method as you type in the program.

This program should run on any computer with

Microsoft BASIC. It was written on an Apple and has two machine-dependent lines. Line 40 clears the screen and vertically tabs down the screen 10 lines. Line 340 clears the screen and homes the cursor. You should substitute the clear screen command for your computer in those two lines. The lower case is purely for cosmetic reasons and, if you have only upper case, then that's what you will get. The REMarks may be ommitted.

Octal numbers are halfway between hexadecimal numbers and binary numbers. They are easier to use than decimal numbers for binary thinking, but they are not commonly used. The standard notation for octal numbers is ö (with two dots above), not always found on computer terminal keyboards. I assigned them the # sign so that the program can tell the numbers apart.

When entering numbers into the program to be converted, the program assumes all numbers to be decimal, unless you prefix the number with a special sign — $ for hexadecimal, % for binary, and # for octal. The program does no checking for proper input; you will get some very strange results if you input illegal numbers.

Hopefully, the next time you see a binary or hexadecimal number, you will understand what they are all about.

Program Description

[10] DIMensions the arrays to store the individual digits of the numbers.

[20-30] Set up the functions to get integer divisions and remainders.

[40-90] Present the introductory screen information and prompt for the number to be converted. A < return > quits the program.

[110, 170, and 230] check to see what type of number you entered.

[120] Converts octal to decimal.

[180] Converts binary to decimal.

[240] Converts hexadecimal to decimal.

[350] Converts negative decimal to positive decimal.

[370] Converts decimal to hexadecimal.

[440] Converts decimal to octal.

[530] Converts decimal to binary.

[710-790] Prints the results and waits for a return to start over.

[800] A subroutine to convert numbers larger than 9 into the A-F hexadecimal letters.

[840] A subroutine to divide A by N, assign Q(I) the integer division result, and return with A equal to the remainder.

[860] A subroutine to assign Q$() and Q() arrays each digit of the input number.

Number Conversion Listing

```
10  DIM Q(20),Q$(20)
20  DEF  FN A(X) =  INT (X / N): REM
      Int function
30  DEF  FN B(X) = X -  FN A(X) *
     N: REM  Mod function
40  HOME : VTAB 10
50  PRINT "This program converts
     numbers into other bases."
60  PRINT "Input your number in t
     he following form:"
70  PRINT "<DECIMAL> or <-DECIMAL
     >,<$HEXIDECIMAL>,"
80  PRINT "<#OCTAL> and <%BINARY>.
90  PRINT : INPUT A$: IF  LEN (A$
     ) = 0 THEN  END
100 A =  VAL (A$)
110  IF  LEFT$ (A$,1) <  > "#" THEN 170
120  REM  Convert Octal to Decimal
130 A$ =  RIGHT$ (A$, LEN (A$) - 1)
140  IF  LEN (A$) < 6 THEN A$ = "
     0" + A$: GOTO 140
150 N = 6: GOSUB 860
160 A = Q(1) * 32768 + Q(2) * 409
     6 + Q(3) * 512 + Q(4) * 64 +
     Q(5) * 8 + Q(6): GOTO 340
170  IF  LEFT$ (A$,1) <> "%" THEN 230
180  REM  Convert Binary to Decimal
190 A$ =  RIGHT$ (A$, LEN (A$) - 1)
200  IF  LEN (A$) < 16 THEN A$ =
     "0" + A$: GOTO 200
210 N = 16: GOSUB 860
220 A = Q(1) * 32768 + Q(2) * 163
     84 + Q(3) * 8192 + Q(4) * 40
     96 + Q(5) * 2048 + Q(6) * 10
     24 + Q(7) * 512 + Q(8) * 256
      + Q(9) * 128 + Q(10) * 64 +
     Q(11) * 32 + Q(12) * 16 + Q(
     13) * 8 + Q(14) * 4 + Q(15) *
     2 + Q(16): GOTO 340
230  IF  LEFT$ (A$,1) <> "$" THEN 340
240  REM  Convert Hex to Decimal
250 A$ =  RIGHT$ (A$, LEN (A$) - 1)
260  IF  LEN (A$) < 4 THEN A$ = "
     0" + A$: GOTO 260
270 N = 4: GOSUB 860
280  FOR I = 1 TO 4
290  IF Q$(I) < "A" THEN 310
300 Q(I) =  ASC (Q$(I)) - 55: GOTO 320
310 Q(I) =  VAL (Q$(I))
320  NEXT
330 A = Q(1) * 4096 + Q(2) * 256 +
     Q(3) * 16 + Q(4)
340  HOME
350  IF A < 0 THEN A = 65536 + A
360 ASAAVE = A
370  REM   Convert Decimal to Hex
380 N = 4096:I = 1: GOSUB 840
390 N = 256:I = 2: GOSUB 840
400 N = 16:I = 3: GOSUB 840
410 Q(4) = A
420 N = 4: GOSUB 800:H$ = A$
430 A = ASAAVE
440  REM  Convert Decimal to Octal
450 N = 32768:I = 1: GOSUB 840
460 N = 4096:I = 2: GOSUB 840
470 N = 512:I = 3: GOSUB 840
480 N = 64:I = 4: GOSUB 840
490 N = 8:I = 5: GOSUB 840
500 Q(6) = A:O$ = ""
510 N = 6: GOSUB 800:O$ = A$
520 A = ASAAVE
530  REM  Convert Decimal to Binary
540 N = 32768:I = 1: GOSUB 840
550 N = 16384:I = 2: GOSUB 840
560 N = 8192:I = 3: GOSUB 840
570 N = 4096:I = 4: GOSUB 840
580 N = 2048:I = 5: GOSUB 840
590 N = 1024:I = 6: GOSUB 840
600 N = 512:I = 7: GOSUB 840
610 N = 256:I = 8: GOSUB 840
620 N = 128:I = 9: GOSUB 840
630 N = 64:I = 10: GOSUB 840
640 N = 32:I = 11: GOSUB 840
650 N = 16:I = 12: GOSUB 840
660 N = 8:I = 13: GOSUB 840
670 N = 4:I = 14: GOSUB 840
680 N = 2:I = 15: GOSUB 840
690 Q(16) = A
700 N = 16: GOSUB 800:B$ = A$
710  PRINT "Decimal="
720  PRINT ASAAVE"   ("ASAAVE - 65536")"
730  PRINT : PRINT "Hexadecimal="
740  PRINT H$
750  PRINT : PRINT "Octal="
760  PRINT O$
770  PRINT : PRINT "Binary="
780  PRINT B$
790  PRINT : PRINT : INPUT "Press
      <return> ";A$: GOTO 40
800 A$ = "": FOR I = 1 TO N
810  IF Q(I) > 9 THEN C$ =  CHR$
     (Q(I) + 55): GOTO 830
820 C$ =  STR$ (Q(I))
830 A$ = A$ + C$: NEXT : RETURN
840 Q(I) =  FN A(A):A =  FN B(A)
850  RETURN
860  FOR I = 1 TO N
870 Q$(I) =  MID$ (A$,I,1)
880 Q(I) =  VAL (Q$(I))
890  NEXT : RETURN
```

LEARNING CENTER

A Beginner's Computer Glossary

Mnemonic — A technique intended to assist human memory; an abbreviation or acronym that is easy to remember. A symbolic representation (e.g., ADD or SUB).

Modem — Acronym for MOdulator/DEModulator A chip or device that converts data from a form that is compatible with data processing equipment to a form compatible with transmission facilities and *vice versa*. It is often used to interface a digital device to a telephone line.

Module — A device or piece of equipment that is interchangeable with other components.

Monitor — 1. To control operation of several unrelated routines. 2. A black and white or color CRT display.

Mother Board — A circuit board used to connect other processor boards, such as CPU cards, cassette interfaces, and memory cards, to name a few.

Nanosecond — A billionth of a second.

Nesting — Placing a routine or program segment within a larger routine or program segment.

No Operation (NOP) — Tells computer to deliberately leave a blank to allow insertion of data or information at a later time without rewriting.

On Line — A system or device in a system that is controlled by the central processing unit. (Off line means the equipment is not under control of the CPU.)

Operation Code (Op Code) — A command, usually given in machine language.

Optimize — Arranging instructions or data in the storage area so that a minimum amount of machine time is spent accessing the instruction or data.

Port — The entry channel to which a data set is attached. It is in the central computer, and each user is assigned one port.

Part 2

PROM — Programmable Read-Only Memory. Generally, any type of memory not recorded during packaging, but can be programmed in later.

Queue — A line or group of items waiting to be processed..

RAM — Random Access Memory. Provides immediate access to any storage location in memory. Information may be written in or read out quickly.

Register — 1. A device for the temporary storage of one or more words to facilitate arithmetical, logical, or transferral operations. 2. The hardware for storing one or more computer words. 3. A term used to designate a specific computer unit for storing a group of bits or characters.

ROM — Read-Only Memory. A memory that is programmed in during packaging. There are many types of ROMs. Information is stored permanently (or semi-permanently) and is read out, but not altered, in operation.

Routine — 1. A sequence of machine instructions. 2. A set of coded instructions in proper sequence that tells the computer to perform an operation or series of operations.

"Smart" terminal — A rudimentary smart terminal consists of a CRT, keyboard, serial communication I/O device, and a microcomputer. It may use peripheral memory devices such as a tape cassette. A "smart" terminal provides built-in capability not alterable by the user; an "intelligent" terminal is user programmable.

Subroutine — A program that defines operations and which may be included in the main routine.

Text Editor — Facilities designed into a computer program to allow keyboarding of text without a format. Once placed in storage, it can be edited and justified to the required specifications.

Variable — A symbol whose numeric value changes from one repetition of a program to the next, or changes within each repetition of a program.

MICRO

Analysis of Bond Quotations on the APPLE

by David C. Lewis

A program to compute information regarding the performance of bonds. Data for computations is available in financial sections of many newspapers.

The bond-analysis program presented here grew out of the realization that I was spending a lot of time with my calculator and a sheet of paper, punching buttons and making notes. It occurred to me that it should be possible to develop a program that prompts the inputs required, does the calculations quickly without any superfluous intervention on my part, and presents the data intelligently. This program meets my requirements.

Three things are necessary to understand this program: you should know a little about the bond market and bond market quotations, you should understand some (by no means all) of the basic criteria that are used to analyze bonds, and you should know quite a lot about string-handling operations. This last item may surprise many microcomputerists. However, for reasons I will explain later, analyses of bond market quotations rapidly become a case study in string-handling procedures.

Financial Background

Before getting into a program to analyze bonds you should understand what bonds are, and what the quoted numbers relating to bonds are. These basic concepts and data are used to develop an approach to the analysis of any particular bond and the bond-analysis program. Basically bonds are a type of promissory note or IOU that many corporations, municipalities, and state and local governments use to finance their projects. To get a feel for the numbers and diversity of bonds simply turn to the financial page of your local newspaper; generally you will see only the corporate bonds that traded recently (i.e., yesterday in a daily paper). Municipal, state, and federal government bonds often are not reported, and corporate bonds that are not bought or sold on any day are not reported that day.

Typically, bonds are issued in units of $1,000. The issuer promises to pay the buyer a fixed percentage of the face value of the bond each year until some date in the future, at which time the issuer will redeem the face value of the bond. Thus, a 10% bond of 1983 would yield its buyer $100 in 1982 and he would get back the full amount ($1,000) in 1983.

Although a bond may have a nominal value of $1,000, its actual price may fluctuate substantially. While no one should pretend to understand all the factors that make the bond market go up or down, many people think that prevailing interest rates have a pronounced effect on the market. Thus, to induce a prospective bond buyer to actually buy a bond, that bond must offer the investor a return on his investment comparable to what he could realize by putting his money elsewhere. If someone purchased a 30-year bond in 1960 that yielded 7%, he would receive $70 per year from the bond. If it were necessary to sell that bond in a market where investors could routinely get 14% on their investments, the original buyer would have to reduce the sales price to $500 so the buyer would realize 14% on his purchase. If the buyer couldn't get 14% then he wouldn't buy the bond. Of course, this line of reasoning would not apply if the bond came due in the next few years, since the buyer could anticipate getting $1,000 in return for whatever he paid for the bond. For example, if someone paid $800 today for a bond that came due in 1983, then the buyer would realize a profit of $200 in 1983, or a return of 25% on his investment. Thus, it is possible to make (or lose) money on bonds in two ways — from the interest payments and from capital appreciation or depreciation.

Bond Quotations
requires:
Microcomputer with
Microsoft BASIC

Often I have heard my broker speak of the "yield-to-maturity" of a bond. This is the sum of the yield on a bond due to its interest and the capital appreciation portion of the bond. If a 10% bond came due in 1990 and was currently selling for $500, then the yield related to the interest income is ($1,000*10%/$500 = 20% and the pro-rated capital appreciation on the bond is ($1,000 - $500)/((1990 - 1982)*$500) = 5.5%, so the yield-to-maturity is 20% + 5.5% = 25.5%.

Yields-to-maturity can be misleading since they include two different types of yields. The interest income is available at least yearly and can be reinvested and compounded. The yield due to capital appreciation, however, is prorated straight line from now to maturity — there is no compounding. For example, if a bond were bought for $100 and matured 20 years later for $1,000, the prorated yield due to

capital appreciation is (($1,000 – $100)/($100*20 years)) = 45% per year. However, if the bond were to pay roughly 12% interest each year, and if that interest could be compounded without taxes, the investor would realize the same capital appreciation. This subtlety is particularly important in analyzing zero-coupon bonds. These bonds generally are sold at much less than face value and pay no annual interest. *All* of the yield on coupon bonds is a result of capital appreciation. When comparing zero-coupon bonds and other types of investments, it is important to consider the yield on a zero-coupon bond (or any capital accumulation yield) and some type of "deflated" basis in which the lack of opportunity to compound your earnings is factored.

Financial Calculations

The program here prompts user inputs and accepts inputs as they are typically published in the literature (i.e., fractional numbers are accepted for price and interest, and the true price is computed based on the price quoted in the newspaper. The program computes the following indices:

1. Interest paid per year in dollars.
2. Number of years from the current year to the year of the bond's maturity.
3. Simple yield based on interest.
4. Yield due to straight-line capital appreciation (i.e., no compounding).
5. Straight-line yield to maturity (i.e., the sum of items 3 and 4 above).
6. The equivalent yield if the capital appreciation could be compounded.
7. Compounded yield to maturity (i.e., the sum of items 3 and 6 above).

Finally, the program presents an annotated listing of each of the seven items listed above and offers the option of providing a hard copy of the program output.

Getting the Data

A principal resource for data regarding bonds is your newspaper. If you look in the financial section of your paper, generally you will see a statement such as:

XYZ INC 9 5/8 02 61 1/2

This means the particular bond issue put out by XYZ Inc. has a yield of 9 5/8% on the face value of the bond ($1,000), will be redeemed in the year 2002 ("02"), and was bought for $615. Note that the quoted price is a factor 10 times smaller than the price shown in the newspaper (i.e., 61 1/2), and interest and price quotations are typically (although not always) given in fractions of 1/8, and only the last two digits of the year of redemption of this or any other bond is quoted. Clearly, some massaging of the input data is necessary before the computer can compute the various yields, dates, returns, etc.

Programming Considerations

The main problem associated with developing the program was creating some mechanism to allow the user to input data as it is typically quoted. Microcomputerists familiar with DATA and INPUT statements know they accept either strings or decimal numbers; Apple will not immediately understand numbers like 9 5/8. To get a microcomputer to accept and manipulate what might be referred to as "fractional numbers" it is necessary to input the data as a string and develop a way to evaluate that string.

The subroutine developed to evaluate the string inputs is shown in figure 1. The routine is structured to interpret a string by first evaluating the denominator of the fractional number, then the numerator, then the integer, and using that information to compute the type of decimal number with which the computer can deal. If no fraction is sensed for a number (i.e., if no "/" is sensed) then the string is evaluated as a number. This option is necessary since bond data is sometimes quoted in integer and even decimal form.

The subroutine uses virtually all the string-handling operations available in MicroSoft BASIC. Apart from some variable setting operations, the first step in the routine is to determine the number of characters (called N) in the string using the LEN( ) statement. Subsequently, each character in the string is examined, starting with the rightmost character, to see if it is a "/". To break out each character in the string the MID$( ) statement is used where N, the number of characters in I$, is obtained by counting from the right of the string, and 1 indicates that PI$ is only 1 character. To determine whether or not the character being examined is a "/", it is compared to the ASCII representation of "/" (i.e., CHR$(47)). If a "/" is sensed, then the program knows it is examining a fraction, and that the MID$(I$,N,1) statement has stepped its way from right to left across the denominator. To sense the value of the denominator, the program simply backs the MID$(I$,N,x) up one character and defines a new string from that character to the right end of the string using the RIGHT$ (I$,N – 1) statement. Then it takes the VAL( ) of that substring to get a real number for the denominator.

To get a real number for the numerator the MID$(I$,1,N) statement is used to search the string for a "space." Thus, you expect a space between the integer portion of the fractional number and its fractional portion. As with the search for the "/", the ASCII representation of each character in the string is compared with CHR$(32), the ASCII representation for a space. When CHR$(32) is sensed, the MID$(I$,N,2) statement has stepped to the beginning of the numerator of the fractional input. To get the value of the numerator, a new substring of I$ is defined that includes the entire fractional portion of I$ and then takes the VAL( ) of the substring. Since the VAL statement evaluates the string up to the first non-numerical character (in this case the "/"), what is returned is the numerator of the fraction in the string. The fractional part of the input string is evaluated by dividing the numerator of the denominator. It's that simple.

One problem was getting the computer to accept and manipulate fractions.

Evaluating the integer portion of the input string is straightforward when you know which character constitutes the start of the numerator. Simply establish a new string, starting from the left, and use the LEFT$( ) statement, whose length is the difference between the length of the input string and the string position of the first digit in the numerator. Then take the value, using the VAL( ) statement, of the new sub-string. If you simply take the VAL( ) of the input string you will get some strange number that includes the integer and numerator characters. For example, if the VAL statement were used on the string 57 3/8, the computer

would read 573 (i.e., all numbers up to the first non-numeric character, skipping over spaces).

Once the integer and fractional portions of the input string have been evaluated, it is easy to develop a number the computer can use — just add the two numbers.

The subroutine described above will evaluate fractional data inputs. To complicate life, bond interest and price data is sometimes given in integer or decimal form. The subroutine deals with this contingency by determining whether or not it finds a "/"; if none is found after stepping across the input string, the program evaluates the number using VAL( ) on the entire input string.

Another programming problem, which also involves strings, relates to the formatting of the output display. The quantities that are calculated by the program are routinely calculated and displayed to nine significant figures. However, there is generally no reason to evaluate a bond's performance to more than three or four significant figures. Displaying all the significant digits adds little to the utility of the program and can make the results harder to read and understand. For example, if the number of significant digits displayed can be limited, it is possible to get the results of the calculations all on the same line as the captions, thus improving the readability of the display.

I limited the number of significant digits displayed by converting the numerical results of the calculations to strings, using the STR$( ) statement, and then using the LEFT$( ) statement to take the four most significant figures. This simple approach is *not* a rounding operation; rather, it is a truncation.

The Bond Program

The program is designed to accept data in the sequence data generally appears in financial periodicals, and also in the formats that are commonly used (i.e., fractional numbers).

The interest is computed in dollars, paid per year, and assumes the bond has a face value of $1,000. Thus, the interest is $1,000 times the interest rate.

The total capital yield is simply the difference between the value of the bond at maturity and its purchase price divided by the purchase price. To get a prorated portion of this yield simply divide the total capital yield by the number of years to maturity. This calculation assumes that the price of a bond will steadily approach its mature value on a straight-line basis; it makes no allowance for market conditions.

The program computes a "net" yield by summing the yields due to interest payments and the yield attributed to the prorated capital appreciation of the face value of the bond. As noted previously, these are two rather different yields since "yield-to-maturity,' while often quoted, is of questionable significance.

There is a fundamental difference between annual interest payments and the prorated straight-line yield that might be attributed to capital appreciation. In particular, the capital appreciation cannot be compounded. To get a better estimate of the yield that can be attributed to capital appreciation, compute the equivalent annual yield that, if compounded, would offer the same net capital appreciation as the simple uncompounded capital yield discussed above. This yield is always less than the uncompounded capital yield.

Next sum the equivalent compounded capital yield and the interest payments to give a more realistic yield-to-maturity.

After completing the calculations outlined above, you may want to make a hard copy of the results, complete another analysis, or quit. The program is set up for an MX-80 operating with a GRAPPLER. The printer portion of the program may have to be adapted for different printers.

Dave Lewis is a scientific project officer in the Department of Navy's Office of Naval Research. He manages a variety of electronic warfare and surveillance programs, when he is not trying to beat the bond market. You may contact Mr. Lewis at 7417 Westwood Park Lane, Falls Church, VA 22046.

```
10  GOTO 390
30  REM
40  PRINT ****BONDS****BONDS****
     BONDS****: RETURN
60  REM
80 I = 0:N =  LEN (I$): IF N =  <
     1 GOTO 350
110  FOR Q = 1 TO N - 1:V = N - Q
     :PI$ =  MID$ (I$,V,1): IF PI
     $ =  CHR$ (47) GOTO 220
150  IF PI$ =  CHR$ (32) GOTO 270

160  REM
170  NEXT : IF I = 0 GOTO 350
190  REM
200  RETURN
220  REM
230 Z$ =  RIGHT$ (I$,Q):A =  VAL
     (Z$): GOTO 160
270  REM
280 B =  VAL ( RIGHT$ (I$,Q)):FI =
     B / A:DI = N - Q:D$ =  LEFT$
     (I$,DI):I =  VAL (D$):I = I +
     FI: GOTO 160
350  REM
360 I =  VAL (I$): GOTO 190
390  REM
400  HOME : GOSUB 30: VTAB 5: PRINT
     ENTER DAY DATE  ;: INPUT D
     D: PRINT ENTER MONTH(1 OR 2
      DIGITS)  ;: INPUT MD: PRINT
     ENTER YEAR(LAST 2 DIGITS);
     : INPUT YD: HOME : GOSUB 166
     0: GOSUB 30
560  REM
580  VTAB 5: PRINT NAME OF BOND
      ;: INPUT TI$: PRINT ENT
     ER INTEREST ;: INPUT BI$:I$
      = BI$: GOSUB 60:BI = I / 10
     0: PRINT ENTER YEAR OF MATU
     RITY(2 DIGITS) ;: INPUT YM$
     :YM =  VAL (YM$): PRINT ENT
     ER BOND PRICE ;: INPUT PR$:
     I$ = PR$: GOSUB 60:PR = I: HOME
     : [illegible] 1660: GOSUB 30: VTAB
     5: PRINT TI$;: PRINT    ;:
      PRINT BI$;: PRINT    ;: PRINT
     YM$;: PRINT     ;: PRINT
     PR$
950  PRINT  :D = 1000 * BI: PRINT
     DOLLARS PAID/YEAR = $;: PRINT
     D:DY = YM - YD: IF DY =  > 0
      GOTO 1060
1050 DY = 100 + DY
1060  PRINT YRS TO MATURITY=;: PRINT
     DY:Y = (BI / PR) * 10000:Y$ =
      STR$ (Y):Y1$ =  LEFT$ (Y$,4
     ):Y1 =  VAL (Y1$): PRINT YI
     ELD=;: PRINT Y1;: PRINT %
     :YTM = ((100 - PR) / (PR * D
     Y)) * 100:T$ =  STR$ (YTM):T
     A$ =  LEFT$ (T$,4):T1 =  VAL
     (TA$): PRINT CAP YLD PER YR
     =;: PRINT T1;: PRINT %:TY
      = Y + YTM:T$ =  STR$ (TY):T
     1$ =  LEFT$ (T$,4):T1 =  VAL
     (T1$)
1320  PRINT YIELD TO MATURITY=;
     : PRINT T1;: PRINT %:ZZ =
     100 / PR:Z1 = ( LOG (ZZ)) /
     DY:Z2 = (( EXP (Z1)) - 1) *
     100:Z2$ =  STR$ (Z2):Z3$ =  LEFT$
     (Z2$,4):Z3 =  VAL (Z3$): PRINT
     COMPOUND CAPTL YLD=;: PRINT
     Z3;: PRINT %:CY = Y + Z2:Y
     $ =  STR$ (CY):Y1$ =  LEFT$
     (Y$,4):Y1 =  VAL (Y1$): PRINT
     COMPOUNDED YIELD TO MATURIT
     Y=;: PRINT Y1;: PRINT %: VTAB
     20
1570  REM
1580  PRINT WANT TO CONTINUE?  Y
     ES/NO/PRINT;
1590  INPUT C$:Z$ =  LEFT$ (C$,1)
     : IF Z$ =  CHR$ (89) GOTO 17
     70
1620  IF Z$ =  CHR$ (80) GOTO 183
     0
1630  IF Z$ =  CHR$ (78) GOTO 189
     0
1640  PRINT ?;: GOTO 1590
1660  REM
1680  HTAB 30: PRINT DD;: PRINT
     /;: PRINT MD;: PRINT /;: PRINT
     YD: RETURN
1770  HOME : GOSUB 30: GOSUB 1660
     : GOTO 560
1830  PRINT PR#1: PRINT S: PRINT
     PR#0: GOTO 1570
1890  HOME : VTAB 12: HTAB 14: PRINT
     FINIS: END
```

EAGLE

Call on Eagle 8 Bit & 16 Bit Computers and Software

NEC

COMPUTERS

6000	CALL
8001A	$719.00
8031	$719.00
8012	$549.00

PRINTERS

8023	$469.00
7710/7730	$2299.00
3510/3530	$1549.00

MONITORS

JB-1260	$119.00
JB-1201	$149.00
JC-1212	$299.00
JC-12-202	$299.00
JC-1203	$599.00

PRINTERS

SMITH CORONA

TP 1	$599.00
Tractor Feed	$129.00

C. ITOH (TEC)

Starwriter(F10-40CPS)	$1299.00
Printmaster(F10-55CPS)	$1649.00
Prowriter 8510 P.	$399.00
Prowriter 8510 S.	$599.00
Prowriter 1550 P	$769.00
Prowriter 1550 S.	$799.00

OKIDATA

82A	$429.00
83A	$659.00
84 (Parallel)	$1049.00
84 (Serial)	$1149.00
92	$599.00
93	$999.00

IDS

MicroPrism	$649.00
132 (Fully Configured)	$1599.00
80 (Fully Configured)	$1399.00

Call for other configurations.

STAR

Gemini 10	$379.00
Gemini 15	$489.00

DAISYWRITER

Letter Quality	$1049.00

DIABLO

620	$999.00
630	$1769.00

TeleVideo

TERMINALS

910	$579.00
912C	$699.00
920C	$749.00
925C	$749.00
950	$950.00
WYSE WY100	$749.00

COMPUTERS

800A	$1299.00
802	$2649.00
802H	$4695.00
806	$4999.00
816	$8999.00
803	CALL
1602/1603	CALL

PANASONIC

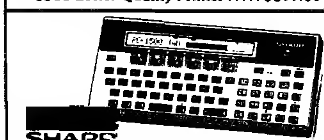
JR200U 32K Pers. Computer	$309.00

MONITORS

TR-120, 12" Hires Green	$159.00
CT-160, 10" Dual Mode Color	$299.00
DT-D1000, 10" RGB	$349.00
DT-D1300, 13" RGB/Composite	$429.00

SANYO

MB 1000 Computer	$1599.00
MB 160 Add on Drive	$539.00
5500 Letter Quality Printer	$699.00



SHARP PC-1500

POCKET COMPUTER

$169.

PC 1250...$89.00

CE 150 Printer, Plotter and Cass. Interface Unit	$172.00
CE 152 Cass. Recorder	$62.00
CE 155 8K Ram Expansion Module	$94.00
CE125 Printer/Micro Cassette	$129.00

MONITORS

AMDEK

300G	$159.00
300A	$169.00
310G	$179.00
310A	$169.00
Color I	$299.00
Color II	$599.00
Color II A	$799.00
Color III	$349.00

BMC

12AU 12" Green	$79.99
1401 13" Color (Mid Res.)	$369.00
9191U 13" Composite	$329.00

TAXAN

Color Composite	CALL
RGB 1	$329.00

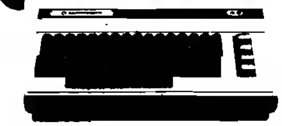
ZENITH

ZVM 121	$99.00
ZT-1 Terminal	$369.00

SHARP

13" Color TV	$269.00
19" Color TV	$339.00

commodore



VIC 20 $149 **VIC 64 $399.**

Motor Mouse	$23.00
Centipede	$23.00
Frogge (VIC)	$23.00
Frogge (64)	$23.00
VIC 20 Dust Cover	$9.99
VIC 1530 Datassette	$69.00
VIC 1541 (64K Disk Drive)	$339.00
VIC 1525 Graphic Printer	$339.00
VIC 1210 3K Mem. Exp.	$32.00
VIC 1110 8K Mem. Exp.	$53.00
VIC 1111 16K Mem. Exp.	$94.00
VIC 1011 RS232C Term. Interface	$43.00
VIC 1112 IEEE-488 Interface	$86.00
VIC 1211 Super Expander	$53.00
VIC Mother Board	$99.00

HES, UMI, EPYX & Creative Software for VIC. Now In Stock!!

PROFESSIONAL SOFTWARE

Word Processing for VIC 64	$79.95
8032	$1039.00
4032	$749.00
8096 Upgrade Kit	$369.00
Super Pet	$1499.00
2031	$469.00
8250 Dbl.Sided Disk Drive	$1699.00
D9060 5 Meg. Hard Disk	$2399.00
D9060 7.5 Meg. Hard Disk	$2699.00
8050	$1299.00
4040	$969.00
8300 (Letter Quality)	$1549.00
8023	$599.00
4022	$399.00
New Z-Ram, Adds CP/M® & 64K	$549.00
The Manager	$209.00
Magis	CALL
Word Pro 5 Plus	$319.00
Word Pro 4 Plus	$299.00
Word Pro 3 Plus	$199.00
The Administrator	$379.00
Info Pro Plus	$219.00
Power	$79.00

HEWLETT PACKARD

41CV $209

HP75 $799.

HP 41C(free memory module)	$149.00
HP 10C	$59.00
HP 11C	$72.00
HP 12C	$99.00
HP 15C	$99.00
HP 16C	$99.00

MODEMS

HAYES

Smart	$219.00
Smart 1200 (1200 Baud)	$549.00
Chronograph	$199.00
Micromodem 100	$309.00
Micromodem II	$279.00
Micromodem II (with Term)	$299.00

NOVATION

J-Cat	$119.00
Cat	$144.00
D-Cat	$159.00
103 Smart Cat	$189.00
Apple Cat II	$279.00
103/212 Smart Cat	$439.00
212 Apple Cat II	$609.00
Apple Cat II 212 Upgrade	$309.00

ANCHOR

Mark I (RS-232)	$79.00
Mark II (Atari)	$79.00
Mark III (TI-99)	$109.00
Mark IV (CBM/PET)	$125.00
Mark V (OSBORNE)	$95.00
Mark VI (IBM-PC)	$179.00
Mark VII (Auto Answer Call)	$119.00
TRS-80 Color Computer	$99.00
9 Volt Power Supply	$9.00

TIMEX SINCLAIR

1000 $85.

16K Memory Module	$44.95
Vu-Calc	$17.95
Check Book Manager	$13.95
The Organizer	$14.95
The Budgeter	$13.95
Stock Option	$14.95
Loan & Mortgage Amortizer	$12.95
Mindware Printer	$109.00
Orbit Software	CALL

IBM®

NEC

3550 PRINTER...$1999

PERCOM/TANDOM DRIVES

5¼" 160K Disk Drive	$249.00
5¼" 320K Disk Drive	$299.00

AMDEK

310A Amber Monitor	$169.00
310G	$179.00
Amdisk (3¼" Drive)	$679.00
XY Plotter	$649.00
Color II	$599.00

SOFTWARE

I.U.S. Easywriter II	$249.00
I.U.S. Easyspeller	$129.00
Peach Package (GL/AP/AR)	$419.00

PROFESSIONAL SOFTWARE

IBM/PC Word Processing	$319.00

CONTINENTAL SOFTWARE

The Home Accountant Plus	$119.00
1st Class Mail/Form Letter	$99.00

SYNAPSE

File Manager	$119.00

Dept. 415

computer mail order east

800-233-8950

IN PA. CALL (717)327-9575, 477 E.THIRD ST., WILLIAMSPORT, PA. 17701

No risk, no deposit on C.O.D. orders. Pre-paid orders receive free shipping within the UPS Continental United States Delivery Zone with no waiting period for certified checks or money orders. Add 3% (minimum $3.00) shipping and handling on all C.O.D. and credit card orders. Larger shipments may require additional charges. NV and PA residents add sales tax. All items subject to availability and price change. **NOTE:** We stock manufacturer's and third party software for most all computers on the market. Call today for our new catalog.

Circ

FRANKLIN

Call on FRANKLIN Computers, Disk Drives, Software and System Specials.

MICRO-SCI

DISK DRIVES FOR APPLE & FRANKLIN

A2	$299.00
A40	$349.00
A70	$459.00
C2 Controller	$79.00
C47 Controller	$89.00

VISICORP

for Apple, IBM & Franklin

Visidex	$189.00
Visifile	$189.00
Visiplot	$159.00
Visiterm	$89.00
Visitrend/Plot	$229.00
VisiSchedule	$229.00
Desktop Plan	$189.00
Visicalc(AppleII+,CBM,IBM)	$179.00

Visicorp prices for IBM may vary slightly.

CONTINENTAL

Home Accnt. (Apple/Atari)	$59.00
The Tax Advantage(Apple,Atari)	$45.00
1st Class Mail/Form Letter(Apple)	$79.00
The Book of Apple	$14.95
The Book of Atari	$14.95
The Book of Apple Graphics	$14.95

SIRIUS

Free Fall	$24.00
Beer Run	$24.00
Snake Byte	$24.00
Space Eggs	$24.00
Sneakers	$24.00
Bandits	$28.00

BRODERBUND

Apple Panic	$23.00
David's Magic	$27.00
Star Blazer	$25.00
Arcade Machine	$34.00
Choplifter	$27.00
Serpentine	$27.00

INFOCOM

Deadline(Atari,Apple,IBM)	$35.00
Star Cross	$29.00
Zork I, II, or III	$29.00

MPC

Bubdisk (128K Ram)	$719.00

AXLON

Apple/Franklin 128K Ram	$399.00
Apple/Franklin Ram Disk	$999.00

KRAFT

Apple Joystick	$44.00

ATARI

1010 Recorder	$74.00
1020 Printer	$269.00
1025 Printer	$589.00
830 Modem	$159.00
820 Printer	$259.00
850 Interface	$169.00
CX40 Joy Sticks (pair)	$18.00
CX414 Bookkeeper Program	$119.00
CX419 Bookkeeper Kit	$195.00
CX481 Entertainer Package	$69.00
CX482 Educator Package	$130.00
CX483 Programmer Package	$54.00
CX484 Communicator Package	$344.00
Full Stroke Replacement Keyboard for Atari 400	$119.00

ALIEN

Atari Voice Box	$119.00
Apple Voice Box	$149.00

MEMORY

Axlon 32K Ram	$89.00
Axlon 48K Ram	$139.00
Axlon 128K Ram	$399.00
Intec 32K Board	$74.00
Intec 48K Board	$99.00
Intec 64K Board(400 Only)	$149.00

WICO

Joystick	$24.95
Famous Red Ball	$26.95
Apple Trackball	$59.00
Atari/VIC Trackball	$55.00
Apple Adapter	$16.00

PERCOM

DISK DRIVES FOR ATARI

AT 88-S1	$399.00
AT 88-A1	$299.00
RFD 40-S1	$549.00
RFD 40-A1	$349.00
RFD 40-S2	$889.00
RFD 44-S1	$679.00
RFD 44-S2	$1029.00

RANA DISK DRIVES

Call for price and availability on the new Rana Disk Drives for The Apple and Franklin Computer Systems.

FLOPPY DISKS

MAXELL

MD I (Box of 10)	$32.00
MD II (Box of 10)	$44.00
FD I (8")	$40.00
FD II (8" DD)	$50.00

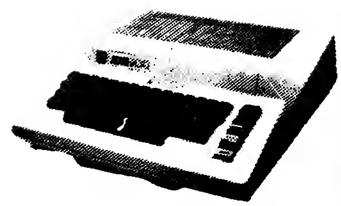
VERSATUM

5¼" SS SD	$26.00
5¼" DS DD	$36.00

ELEPHANT

5¼" SS SD	$19.99

48K 800 $499



ATARI 400

16K	$199
32K	$274*
48K	$299*
64K	$359*

*Non-Atari Ram

One Year Extended Warranty $70.00

810 Disk Drive $429.00

Call for Price and Availability of the NEW

64K ATARI 1200

ATARI

Pac-Man	$33.00
Centipede	$33.00
Caverns of Mars	$32.00
Asteroids	$29.00
Missile Command	$29.00
Star Raiders	$35.00
Galaxian	$33.00
Defender	$33.00
Atari Visicalc	$159.00

ON-LINE

Jawbreaker	$27.00
Softporn	$27.00
Wizard and the Princess	$29.00
The Next Step	$34.00
Mission Asteroid	$22.00
Mouskattack	$31.00
Frogger	$31.00
Cross Fire (ROM)	$36.00

SYNAPSE

File Manager 800+	$69.00
Chicken(Rom)	$34.00
Picnic Paranoia(Rom)	$34.00
Claim Jumper (Rom)	$34.00
Slime (Rom)	$34.00
Shamus(Rom)	$34.00
Protector (Rom)	$34.00
Dodge Racer (C/D)	$26.00
Nautilus (C/D)	$26.00
Shadow World (C/D)	$26.00
Survivor (C/D)	$26.00
Drelbs (C/D)	$26.00
Necromancer (C/D)	$26.00
Pharohs Curse (C/D)	$26.00
Fort Apocolypse (C/D)	$26.00
Page 6	$19.00
Assembler	$30.00
Disk Manager	$24.00

DATASOFT

Pacific Coast Highway	$25.00
Canyon Climber	$25.00
Tumble Bugs	$25.00
Shooting Arcade	$25.00
Clowns and Balloons	$25.00
Graphic Master	$30.00
Graphic Generator	$13.00
Micro Painter	$25.00
Text Wizard	$79.00
Spell Wizard	$64.00
Bishop's Square	$25.00
Sands of Egypt	$25.00
Moon Shuttle	$25.00
Zaxon	$29.00

APX

Text Formatter	$18.50
Family Budgeter	$18.50
Eastern Front	$24.00
Family Cash	$18.50
Jukebox	$13.50
Downhill	$18.50
Outlaw	$18.50
Holy Grail	$24.00
Player Piano	$18.50
Keyboard Organ	$18.50
Number Blast	$13.50
Frogmaster	$18.50
747 Land Simulator	$18.50
Bumper Pool	$13.50

CBS

K-razy Shoot Out	$32.00
K-razy Kritters	$32.00
K-razy Antics	$32.00
K-star Patrol	$32.00
Stick Stand	$5.99

EPYX

Crush, Crumble & Chomp	$24.00
Crypt of the Undead	$24.00
Curse of Ra	$16.00
Datestones & Ryn	$16.00
Invasion Orion	$19.00
King Arthur's Heir	$24.00
Morloc's Tower	$16.00
Rescue at Rigel	$24.00
Ricochet	$16.00
Star Warrior	$29.00
Temple of Asphai	$29.00
Upper Reaches of Apshai	$16.00

SPINNAKER

Snooper Troops #1	$34.00
Snooper Troops #2	$34.00
Face Maker	$24.00
Story Machine	$24.00
Delta Drawing	$45.00
Rhymes and Riddles	$21.00
Kinder Comp	$21.00

ROKLAN

Wizard of War (Rom)	$34.00
Deluxe Invader (Rom)	$29.00
Gorf (Rom)	$34.00

FIRST STAR

Astro Chase	$25.00

BIG 5

Miner 49er	$35.00

GAMESTAR

Baja Buggies	$24.95
Football	$24.95

computer mail order west

Dept. 415

800-648-3311

IN NV. CALL (702)588-5654, P.O. BOX 6689, STATELINE, NV. 89449

ERNATIONAL ORDERS: All shipments outside continental United States must be pre-paid by certified check only! Include 3%(minimum $3.00) shipping and handling.

DUCATIONAL DISCOUNTS: Additional discounts are available from both Computer Mail Order locations to qualified Educational Institutions.

PO & FPO: Add minimum $5.00 shipping on all orders.

CP/M is a registered trademark of Digital Research, Inc.

Mutual Fund Charting

Two programs to make, update, and print mutual fund files on both OSI and Apple Computers

Buy low, sell high! Sound advice for any investor but not easy to achieve. The microcomputer has opened a new avenue for the small investor to quickly store and then easily display collected data in a manner that can assist materially in decision making — a vital requisite to successful investing.

Mutual funds and money market funds are investment vehicles tried by many, including this author. Inspiration for the program described here came from an article in *Creative Computing* (May 1981) that presented a computer-assisted method of investment analysis developed in part by Richard J. Fabian, a registered financial advisor in the state of California. Authors Browning and Clemmens enlarged upon this procedure using a SWTPC 6800 computer system.

A mutual fund tends to follow the rise and fall of stock market averages. A money market fund, on the other hand, remains more stable. Exchanging the stock fund for the money market fund and *vice versa* minimizes the effects of falling prices but takes advantage of rising prices. The investor must follow stock market trends by charting selected mutual funds and stock market averages over a 39-week period. When the current price moves through the 39-week average an exchange signal is generated. That is, when the trend is downward, change to the money market fund; when the trend is upward, change to the mutual stock fund.

Mutual Funds
requires:
OSI with 065D or
Apple II with one drive
Dot-matrix printer needed for hardcopy printouts

The program generated for this article attempts to duplicate the approach used by Browning and Clemmens, but is adapted to the OSI-065D operating system on a C4P-MF and an Epson MX-80 printer. One part of the program, "FILCHG," enables you to update data files for each Friday's closing price or when the fund makes a distribution (dividend and/or capital gains). The other part of the program arranges data from the data files for the printer to display in chart form.

Data files must be established first, then changed later. "FILCHG" provides this option in lines 80-120. The funds and averages I selected are defined as string variables (line 40) to be recalled by either part of the program. To initially set up the data files the latest 39 consecutive weeks of prices are gathered from financial pages of major newspapers. Option A is selected (line 120) and the program loops to line 490 where separate files are identified in a printout to the screen (lines 150-180). The file to be initialized is selected in line 200. The value of X (line 210) assigns the file to be worked to the variable, N$.

Next, control passes to a subroutine at line 690. Thirty-nine numerical items are entered in an array, the file is written (line 440), and program control returns to line 80. A new file to be initialized can be selected again until all files are filled.

Previous and current distributions of the fund must be considered to accurately reflect current price trends. If a distribution occurs during these 39 weeks it must be subtracted from all earlier entries. Note that when a data item is a distribution (see line 730), program control goes to line 840. This subroutine subtracts the entered amount from each previous price, adjusts the loop counters, and returns to the data-gathering loop at line 760. When all data is entered, the file is written (lines 440-470). A new file now can be opened and this process repeated or the program terminated, depending on the choice selected in line 120.

Once files are established, you can update the files each week or at regular intervals by selecting option B (line 120). The program then moves to line 140 for a screen clear and printout of the mutual fund data-file choices (lines 150-200). When a choice is made (line 200), the next statement defines N$ for the subroutine at line 310. Here the disk file is opened and data is taken off the disk and placed into array A.

If the data is a price change, it is entered at lines 390-400 as a string (to accommodate the '/' symbol used for exiting the loop). This string is changed to a number and multiplied by 100 for storing on disk which eliminates trailing zeros. Line 430 discards the oldest data and enters the new data in the 39-item array. The program now loops back to line 400 for a new entry or the exit ('/') symbol. When no new item is to be entered, the 39 data items are returned to disk (line 450). Line 470 prompts the user that the file is closed and waits for any key press, after which the program returns to line 80 where the file choices are displayed again on the CRT. Another file may be updated or the program exited.

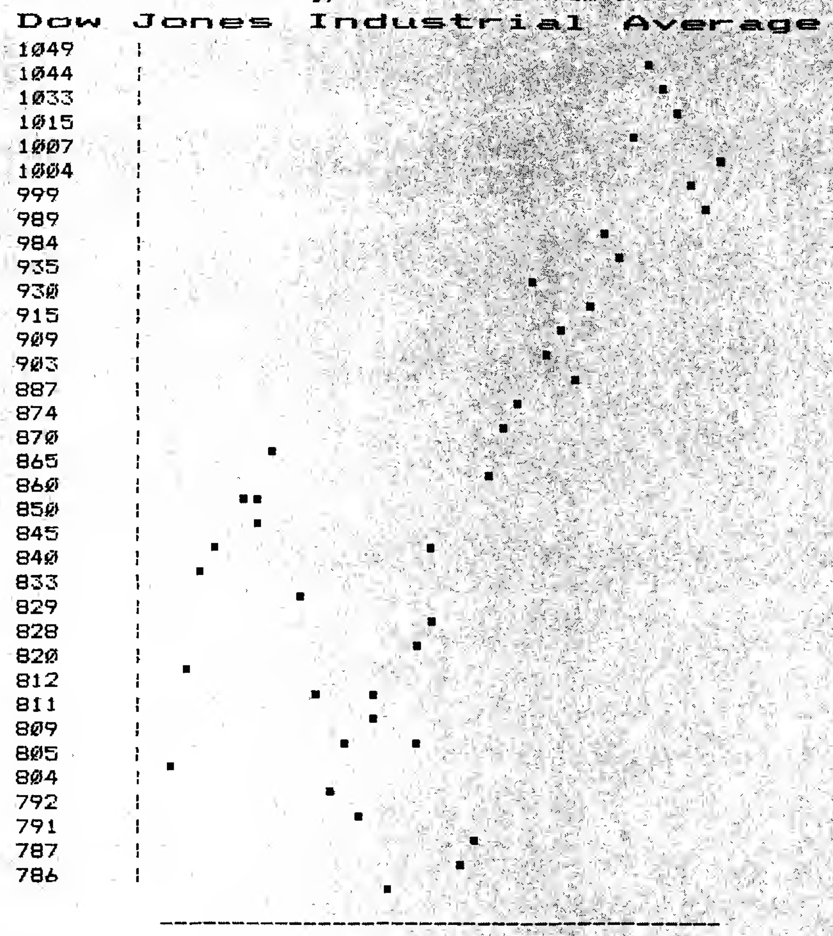
A final note on program "FILCHG" concerns entering a distribution. Line 350 asks if data is a distribution. If so, the amount is entered (line 370), subtracted from all entries in the current

for APPLE and OSI

by Ralph H. Green

***Table 1:* Sample Printout**

```
The following charts are plots of selected Mutual Funds
and selected Stock Market indices for the past 39 weeks.
All points are corrected for distributions made by the
funds during the 39 weeks. All prices are the Friday
closing price.
No investment strategy is indicated from this data.
```




```
39 Week Average = 883.Ø8
Percentage change since last high = 4.17%
Percentage change since last low = 28.12%
Most recent entry is 11/19/82
```

file, and the 39 updated items are returned to disk storage (lines 440-470).

The PRINT program plots the data by providing the necessary commands to the printer. The program uses the CRT as much as possible; the printer is used only to plot the charts. The available choices are printed to the screen (line 1200). One subroutine accomplishes the printing task for all the charts. At line 200, the user inputs a number corresponding to his choice, after which the serial port is activated (line 1410), the corresponding data file is opened (line 310), and the chart heading printed double-size. Line 320 transfers data from the disk and divides by 100 (line 1350), as mentioned earlier. At the same time the data is established in an array (A) and a companion array (B), which has an ascending number that corresponds to the order in which the data appears in the original file.

Since the printer can only advance, not reverse, you must arrange the data in descending order, highest to lowest. A "bubble sort" routine is used for this. Lines 1370-1400 sort the data in descending order with the (B) array number tagging along with its original data item. More on this later.

Line 1410 is a required printer command that sets the proper paper advance. Line 1420 begins the printing of the 39 data items. Lines 1420-1450 ensures that the digits printed at the left of the chart (the share price) have all necessary trailing zeros. CHR$(124) is the vertical line and CHR$(160) is the small square denoting a point on the chart. Lines 1450-1470 plot all points corresponding to a particular price by preventing a line feed until necessary. Also, the numbers in the (B) array are tabs for the print head to place the point at the proper week when the price originally occurred.

After all points are printed, the amount of paper advance is reset to a new value (line 1480) and the chart is underlined (line 1490). Some useful data is then calculated and printed in lines 1500-1570. Since the Dow Jones Industrial averages and the Standard and Poor's 500 stock averages are not computed in dollars, lines 1490-1520 route program control where desired.

With the exception of certain commands peculiar to OSI machines, the programs are written in BASIC easily transportable to other microcomputers.

(Continued on next page)

The commands are:

1. DISK!"IO ,03" and DISK!"IO ,02" to activate and deactivate the serial port.
2. DISK OPEN,6,"file" and DISK CLOSE,6 to open or close a data file.
3. DISK GET,X and PRINT#6 and INPUT#6 and DISK PUT, which are used with both sequential and random file access to and from the disk.
4. The screen clear routine.

Most other computer systems supporting data files and a serial port have appropriate commands to accomplish these tasks.

As Browning and Clemmens stress in their article, the investor should spend at least an hour or more each week updating the files and perusing financial columns in daily newspapers. Especially critical are times when exchange signals might be generated. Using this program does not ensure success, but it does serve as an additional tool for making investment decisions.

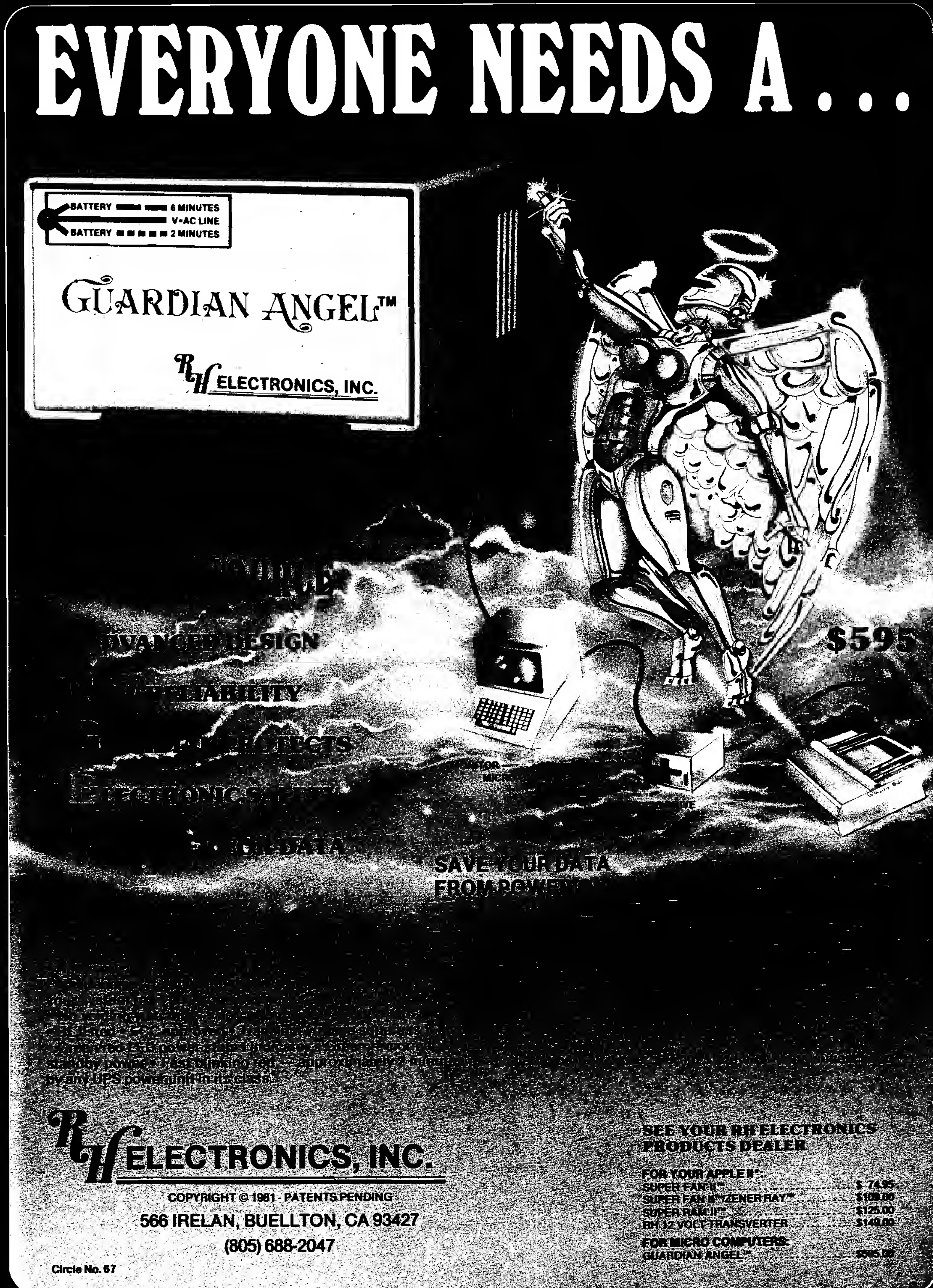
You may contact the author at 2130 16th Street, Greeley, CO 80631.

Listing 1: Mutual Fund Charting

```
10 REM Pgm called FILCHG to upd
   ate Mutual Fund Data Files
20 REM By Ralph Green for OSI
30 REM Translated for Apple ][
   by Philip Daley
40 FOR I = 0 TO 9: READ B$(I): NEXT
70 DIM A(40),B(40):D$ = CHR$ (4)
80 HC = 0: HOME : VTAB 5: PRINT "
   This program enters:": PRINT
90 PRINT "(A) All 39 new values
   in a specified": PRINT "
   Mutual Fund data file, or
100 PRINT "(B) Updates with the
   newest data": PRINT "  and
   discards old items in the file."
110 PRINT "(C) Print out graph o
   f data ": PRINT "   on scre
   en or printer."
115 PRINT "(D) Exit
120 PRINT : PRINT "Which do you
   prefer? ";: GET A$: PRINT A$
   : IF A$ = "A" THEN GOSUB 49
   0: GOTO 80
123 IF A$ = "B" THEN GOSUB 140:
   GOTO 80
125 IF A$ = "C" THEN GOSUB 1000
   : GOTO 80
130 HOME : END
140 C$ = "update": GOSUB 150: GOSUB
   310: GOSUB 340: GOTO 80
150 HOME : VTAB 5: PRINT "You are
   going to "C$" your data file."
170 PRINT : PRINT "Choice for th
   e data file is as follows:"
180 PRINT : FOR I = 0 TO 9: PRINT
   I"-"B$(I): NEXT
200 PRINT : PRINT "What is your
   choice? ";: GET A$:X = VAL
   (A$): IF X = 0 THEN HOME : END
210 N$ = B$(X): RETURN
310 PRINT D$"OPEN"N$: HOME : VTAB
   5: PRINT "You are to "C$" the"
*OSI REM DISK OPEN,6,N$: POKE 12076,3:
   POKE12042,255
311 PRINT N$" file."
320 PRINT D$"READ"N$: FOR I = 1 TO
   39: INPUT A(I): NEXT
*OSI REM INPUT#6,A(I)
330 PRINT D$"CLOSE": RETURN
340 IF X > 6 THEN 390
350 PRINT : PRINT "Do you have d
   istribution information? ";:
   GET A$: PRINT A$
360 IF A$ = "N" THEN 390
370 INPUT "Distribution amount?
   ";A$:Z = VAL (A$): FOR I =
   1 TO 39:A(I) = A(I) - Z * 100
380 NEXT : GOTO 440
390 PRINT : PRINT "Enter new dat
   a item(s), use '/' to end."
400 INPUT "Data item= ";Y$: IF Y
   $ = "/" THEN GOSUB 440: RETURN
410 Y = VAL (Y$): IF X > 6 THEN 430
420 Y = 100 * Y
430 FOR I = 1 TO 38:A(I) = A(I +
   1): NEXT :A(39) = Y: GOTO 400
440 HOME : VTAB 5: PRINT "Now sa
   ving data. Please wait for '
   DONE' prompt."
450 PRINT D$"OPEN"N$: PRINT D$"W
   RITE"N$: FOR I = 1 TO 39: PRINT
   A(I): NEXT : PRINT D$"CLOSE"
*OSI REM DISKGET,J-1:PRINT#6,A(I):DISKPUT:
   NEXT:DISK CLOSE,6
470 PRINT : PRINT "DONE-Press an
   y key to continue.": GET A$:
   PRINT : RETURN
*OSI REM DISK!"GO 252B"
490 HOME : VTAB 5: PRINT "This s
   ection enters all 39 new
500 PRINT "data entries in a spe
   cified file
510 C$ = "enter": GOSUB 150
690 PRINT : PRINT "Enter data fo
   r each of the 39 entries."
700 PRINT "If you have distribut
   ions to enter,
710 PRINT "when the 'Value?' pro
   mpt appears,
720 PRINT "enter 'D'."
730 PRINT :K = 0: FOR J = 1 TO 3
   9: PRINT J" Value ";: INPUT
   Z$: IF Z$ = "D" THEN GOSUB
   840: GOTO 760
740 A(J) = VAL (Z$): IF X > 6 THEN
   760
750 A(J) = A(J) * 100
760 K = K + 1: NEXT : PRINT : GOSUB
   440: RETURN
840 PRINT : INPUT "Diatribution
   amount? ";Z$:Z = VAL (Z$): FOR
   J = 1 TO K
850 A(J) = A(J) - Z * 100: NEXT :
   J = J - 1:K = K - 1: PRINT
860 PRINT "Continue with your en
   tries.": PRINT : RETURN
899 DATA EXIT
900 DATA Fidelity Destiny Fund,
   Oppenheimer Special Fund,Ame
   rican Harbor Fund
901 DATA Sigma Inveatment Share
   s,Investment Company of Amer
   ica,Income Fund of America
902 DATA Dow Jonea Industrial A
   verage,NYSE Common Stock Ind
   ex
903 DATA  Standard & Poor's 500
   Stock Average
1000 REM Pgm called PLOTF
1010 REM Pgm to plot 39 week av
   erage of
1015 REM aelected mutual funds
1020 HOME : VTAB 5: INPUT "Lates
   t date of entries? ";Z$
1030 PRINT : PRINT "Hardcopy? ";
   : GET A$: PRINT A$: IF A$ =
   "Y" THEN PRINT D$"PR#1":HC = 1
*OSI REM DISK!"IO ,03"
1060 PRINT CHR$ (27)"A" CHR$ (1
   33); CHR$ (27)"2"
1070 PRINT "The following charts
   are plots of selected Mutua
   l Funds
1080 PRINT "and selected Stock M
   arket indices for the past 3
   9 weeks.
1090 PRINT "All points are corre
   cted for distributions made
   by the
1100 PRINT "funds during the 39
   weeks. All prices are the Fr
   iday
1105 PRINT "closing price.": PRINT
1110 PRINT "No investment strate
   gy is indicated from this da
   ta.
1180 PRINT D$"PR#0": IF HC = 0 THEN
   GET A$: PRINT
*OSI REM DISK!"IO ,02"
1200 C$ = "print": GOSUB 150: GOSUB
   310
1350 T1 = 0: FOR J = 1 TO 39: IF
   X < 7 THEN A(J) = A(J) / 100
1360 T1 = T1 + A(J):B(J) = J: NEXT
   :A1 = A(39)
1370 R = 0: FOR J = 2 TO 39: IF A
   (J) < = A(J - 1) THEN 1400
1390 R = 1:S = A(J - 1):A(J - 1) =
   A(J):A(J) = S:S = B(J - 1):B
   (J - 1) = B(J):B(J) = S
1400 NEXT : IF R = 1 THEN 1370
1410 IF HC = 1 THEN PRINT D$"PR
   #1": PRINT : PRINT : PRINT CHR$
   (14);N$: PRINT CHR$ (27)"A"
   CHR$ (129); CHR$ (27)"2"
1420 A2 = A(1):A3 = A(39): FOR J =
   1 TO 39:A$ = STR$ (A(J)): IF
   X > 6 THEN 1450
1430 IF INT (A(J)) = A(J) THEN
   A$ = A$ + ".00": GOTO 1450
1440 IF INT (10 * A(J) + .05) /
   10 = A(J) THEN A$ = A$ + "0"

1450 PRINT A$;: POKE 36,8: PRINT
   CHR$ (124);
1460 POKE 36,B(J) + 9: PRINT CHR$
   (27)">" CHR$ (160);
1465 PRINT CHR$ (27)"=";
1470 IF A(J) = A(J + 1) THEN J =
   J + 1: GOTO 1460
1480 PRINT : NEXT : PRINT CHR$
   (27)"A" CHR$ (133); CHR$ (27
   )"2"
1490 FOR J = 1 TO 39: POKE 36,9 +
   J: PRINT "-";: NEXT : PRINT
   : IF X = 8 THEN 1510
1500 IF X > 6 THEN 1520
1510 POKE 36,15: PRINT "39 Week
   Average = $" INT (100 * (T1 /
   39) + .5) / 100: GOTO 1530
1520 POKE 36,15: PRINT "39 Week
   Average = " INT (100 * (T1 /
   39) + .5) / 100
1530 A4 = 100 * (A2 - A1) / A1:A5
   = 100 * (A1 - A3) / A3
1540 POKE 36,15: PRINT "Percenta
   ge change since last high = ";
1550 PRINT INT (100 * A4 + .5) /
   100"%"
1560 POKE 36,15: PRINT "Percenta
   ge change since last low = ";
1565 PRINT INT (100 * A5 + .5) /
   100"
1570 POKE 36,15: PRINT "Most rec
   ent entry is "Z$: PRINT D$"P
   R#0"
1580 PRINT : PRINT "To continue,
   press any key.": GET A$: PRINT
   : RETURN
```

MICRO



EVERYONE NEEDS A . . .
BATTERY 6 MINUTES
V-AC LINE
BATTERY 2 MINUTES
GUARDIAN ANGEL™
RH ELECTRONICS, INC.
$595
SAVE YOUR DATA
RH ELECTRONICS, INC.
COPYRIGHT © 1981 · PATENTS PENDING
566 IRELAN, BUELLTON, CA 93427
(805) 688-2047
Circle No. 67
SEE YOUR RH ELECTRONICS PRODUCTS DEALER
FOR YOUR APPLE II®:
SUPER FAN II™ $ 74.95
SUPER FAN II™ ZENER RAY™ $109.00
SUPER RAM II™ $125.00
RH 12 VOLT TRANSVERTER $149.00
FOR MICRO COMPUTERS:
GUARDIAN ANGEL™ $595.00

LETTERMASK: A Check Protecting Algorithm

by Barton M. Bauers, Jr.

In addition to number masking, this routine creates checks with the amounts spelled out, for additional security.

LETTERMASK
requires:
BASIC

In the August 1980 issue of MICRO (27:65), I discussed the tendency of binary computers to introduce small rounding errors when adding decimal numbers, and proposed a solution that programmers could implement to prevent these errors. In summary, decimal numbers between 0 and 1 cannot be represented exactly in binary mathematics, due to the limitation of precision (the number of places to the right of the decimal point) available to most computers. The solution involved storing all numbers within the program as integer numbers, and 'masking' them on output so they resumed their decimal form when printed. The function that converted the decimal values to integer for internal storage was:

```
DEF FN VL(X) = INT ( (X + .0001) * 100)
```

where X was any real number with two decimal places, and VL(X) was its integer equivalent for internal purposes.

The intent of the article was to preclude this rounding error in handling money calculations, and I included an example of utilizing the subroutine MASK to create check-protection with leading and trailing asterisks (*), as you see so often in computer-generated checks.

Subsequent programming requirements have led me to write a different kind of mask algorithm for check protection — one that spells out the amount when printed, much as you do when you write checks manually. This method is excellent for protecting checks from alteration because the spelled-out values are of varying lengths and are much more difficult to fraudulently change. I consider the word method of check protection preferable to the simple number mask and have created the subroutine 'LETTERMASK' for this purpose. Although most computer-generated checks continue to use some version of the number-masking system (my own still do, in addition to the word masking), I hope the simplicity of the LETTERMASK subroutine will prompt programmers to add this extra protection to check-printing routines.

Almost all numbers can be represented with two sets of words. These are the words 'one, two, three, ..., eight, nine' and the words 'ten, twenty, thirty, ..., eighty, ninety.' I say *almost* all, because there are the numbers from 11 to 19, which, unfortunately, require a separate set of words. This oddity creates some minor programming complications, but it does not make the problem unsolvable.

For purposes of clarity, I refer to the first list (the words one through nine) as Word List A, the second list (the words ten through ninety) as Word List B, and the 'teens' list (11 through 19) as Word List C. In the program, these lists are referred to separately.

Subroutine LETTERMASK properly encodes any value from $.00 to $9,999.99, and returns a word string for that amount. The upper limit is arbitrary and could be changed without too much difficulty. Values below $1.00, and the cents portion of any value, are returned as numbers. In addition, the routine replaces the standard ASCII 0 with the letter O to make the printout of the cents more readable. I recommend that in all check-writing programs, 0's be replaced with O's to spare the bank and the recipient of the check having to decipher the value and, perhaps, from making an error. Many people confuse the number 0 with the number 8 if they are not familiar with the ASCII convention.

The format for the output of the subroutine is:

***[ONE..NINE THOUSAND] [ONE..
NINE HUNDRED] [ONE..NINETY
NINE] DOLLARS AND [00..99]
CENTS***

The input to the subroutine is the variable AMT, which is created in your main program with the value you wish to have printed out as a lettermask. *This value must be an integer number* — no decimal places are to be shown. The subroutine will return with your masked number as variable T$.

Subroutine LETTERMASK works quite simply. First it determines how many digits are in the integer number AMT that you present to the subroutine. Based on that value, one of six branches is taken (lines 20000 through 20040). The program then 'cascades' down from the most significant digit toward the cents part of the value, until the entire number has been converted. Note that REM statements have been used to separate the thousands, hundreds, tens, ones, and teens conversion routines. Using the thousands section (lines 20100 to 20130) as an example, follow the steps the program takes.

The computer evaluates the A*th* element of the variable AMT (in this case A = 1, so it looks at the first, or leftmost, digit). The variable K is set to this value and a branch to line 20700 is taken to get the proper word list from Word List A. The string variable T$, previously loaded with "***", is now

(Continued on page 104)

lengthened with the proper word and the word "THOUSAND".

Hundreds and tens are created similarly, except if the tens digit happens to be a '1'. This means that the value for the tens and ones digits together could be any number from 10 to 19 and, unless the value of the ones digit were 0, the word "TEN" is not appropriate and the 'teens' list is required. A branch is therefore taken to line 20450 to determine whether or not a special word from Word List C is needed.

In the cents section, lines 20500 to 20610, the two rightmost digits are scanned to find any ASCII 0's so they can be converted to the letter O for clarity on printout. Note that at line 20520, if the number of cents is less than 10, then the leading zero is required and the letter 0 is put into variable QQ$. At lines 20540 to 20580, the cents digits are scanned and then added to QQ$. Line 20600 covers the

LETTERMASK

```
10 REM *************************
20 REM *
30 REM *      LETTERMASK
40 REM *  BARTON M. BAUERS JR.
50 REM *
60 REM *************************
70 REM *
80 REM *  RESERVED VARIABLES
90 REM *
100 REM *      REAL
110 REM *  A,  AMT,  K,  J
120 REM *
130 REM *     STRING
140 REM * AMT$, K$, Q$, QQ$, T$
150 REM *
160 REM *************************
170 REM *
180 REM *    READ IN VALUE
190 REM *
200 REM *************************
500 HOME
510 INPUT "ENTER NUMBER ";AMT
520 GOSUB 20000
530 PRINT T$
540 GOTO 510
20000 A = 0:K = 0
20010 AMT$ = STR$ (AMT)
20020 J = LEN (AMT$)
20030 T$ = "***"
20040 ON J GOTO 20500,20500,2040
   0,20300,20200,20100
20050 PRINT "NUMBER TOO LARGE ":
   T$ = "***VOID***": RETURN
20095 REM ***********************
20096 REM *
20097 REM *     THOUSANDS
20098 REM *
20099 REM ***********************
20100 A = A + 1
20110 K = VAL ( MID$ (AMT$,A,1))
20120 GOSUB 20700
20130 T$ = T$ + K$ + " THOUSAND"
20195 REM ***********************
20196 REM *
20197 REM *    HUNDREDS
20198 REM *
20199 REM ***********************
20200 A = A + 1
20210 K = VAL ( MID$ (AMT$,A,1))
20220 GOSUB 20700
20230 IF K$ = "" THEN 20300
20240 T$ = T$ + K$ + " HUNDRED"
```

instance when an amount being converted has cents only and no dollars. Finally, at line 20610, the entire string T$ is completed with the addition of the proper cents mask.

To try subroutine LETTERMASK, type in the following lines of code after saving LETTERMASK to disk. (These lines are not part of the actual subroutine, so they should not be saved to disk.)

```
500 HOME
510 INPUT "ENTER NUMBER" ; AMT
520 GOSUB 20000
530 PRINT T$
540 GOTO 510
```

Type "RUN" and enter some numbers. The computer will print out a properly masked value that provides more safety than the numeric masks commonly used. Remember, all numbers read in must be integers.

When you print T$ on a check, you have to be careful to either omit any other information from that print line, because of the varying length of T$, or you have to set up a method of spacing to allow for the unknown length. One method of doing the latter, if your checks will not permit the balance of the line to be blank, is to use the following convention:

```
xxx PRINT T$; SPC(yy – LEN(T$) );
[Balance of line]
```

xxx refers to your line number, and yy to the distance from the leftmost character of T$ to the leftmost character of the next item you wish to print on the same line. By my calculations LETTERMASK's longest word string is 71 characters.

A final note: Other than checking for a number that exceeds six digits, LETTERMASK does no error checking.

You may contact the author at 30 Hillock Drive, Wallingford, CT 06492.

LETTERMASK *(continued)*

```
20295 REM **********************
20296 REM *
20297 REM *   TENS
20298 REM *
20299 REM **********************
20300 A = A + 1
20310 K = VAL ( MID$ (AMT$,A,1))
20320 GOSUB 20900
20330 IF K = 1 THEN GOTO 20400
20340 T$ = T$ + K$
20395 REM **********************
20396 REM *
20397 REM *     ONES
20398 REM *
20399 REM **********************
20400 A = A + 1
20410 IF K = 1 THEN 20450
20420 K = VAL ( MID$ (AMT$,A,1))
20430 GOSUB 20700
20440 GOTO 20480
20445 REM **********************
20446 REM *
20447 REM *     TEENS
20448 REM *
20449 REM **********************
20450 K = VAL ( MID$ (AMT$,A,1))
20460 IF K = 0 THEN 20480
20470 GOSUB 21100
20480 T$ = T$ + K$ + " DOLLARS AND "
20495 REM **********************
20496 REM *
20497 REM *     CENTS
20498 REM *
20499 REM **********************
20500 K = VAL ( RIGHT$ (AMT$,2))
20510 QQ$ = ""
20520 IF K < 10 THEN QQ$ = "O"
20530 K$ = STR$ (K)
20540 FOR A = 1 TO 2
20550 Q$ = MID$ (K$,A,1)
20560 IF Q$ = "0" THEN Q$ = "O"
20570 QQ$ = QQ$ + Q$
20580 NEXT
20590 K$ = ""
20600 IF J < 3 THEN K$ = " ZERO
   DOLLARS AND "
20610 T$ = T$ + K$ + QQ$ + " CENT
   S***"
20620 RETURN
20621 REM **********************
20622 REM *     END
```

LETTERMASK *(continued*

```
20695 REM **********************
20696 REM *
20697 REM *   WORD LIST A
20698 REM *
20699 REM **********************
20700 ON K GOTO 20720,20730,2074
   0,20750,20760,20770,20780,20
   790,20800
20710 K$ = "": RETURN
20720 K$ = " ONE": RETURN
20730 K$ = " TWO": RETURN
20740 K$ = " THREE": RETURN
20750 K$ = " FOUR": RETURN
20760 K$ = " FIVE": RETURN
20770 K$ = " SIX": RETURN
20780 K$ = " SEVEN": RETURN
20790 K$ = " EIGHT": RETURN
20800 K$ = " NINE": RETURN
20895 REM **********************
20896 REM *
20897 REM *   WORD LIST B
20898 REM *
20899 REM **********************
20900 ON K GOTO 20920,20930,2094
   0,20950,20960,20970,20980,20
   990,21000
20910 K$ = "": RETURN
20920 K$ = " TEN": RETURN
20930 K$ = " TWENTY": RETURN
20940 K$ = " THIRTY": RETURN
20950 K$ = " FORTY": RETURN
20960 K$ = " FIFTY": RETURN
20970 K$ = " SIXTY": RETURN
20980 K$ = " SEVENTY": RETURN
20990 K$ = " EIGHTY": RETURN
21000 K$ = " NINETY": RETURN
21095 REM **********************
21096 REM *
21097 REM *   WORD LIST C
21098 REM *
21099 REM **********************
21100 ON K GOTO 21110,21120,2113
   0,21140,21150,21160,21170,21
   180,21190
21110 K$ = " ELEVEN": RETURN
21120 K$ = " TWELVE": RETURN
21130 K$ = " THIRTEEN": RETURN
21140 K$ = " FOURTEEN": RETURN
21150 K$ = " FIFTEEN": RETURN
21160 K$ = " SIXTEEN": RETURN
21170 K$ = " SEVENTEEN": RETURN
21180 K$ = " EIGHTEEN": RETURN
21190 K$ = " NINETEEN": RETURN
```

MICRO™

Interface Clinic

by Ralph Tenny

In my first article (MICRO 58:108) I presented various hardware and interfacing terms, one of which was "decoder." Functionally, a decoder can be made with a variety of techniques, but the usual approach is to use one or more ICs. The purpose of a decoder is to produce a unique signal that relates to (usually) a memory address appearing on the bus of a microcomputer.

Figure 1 shows a graphic representation of several 16-bit binary addresses like those produced by every instruction cycle of the typical 8-bit microcomputer (such as the 6502 or 6809). The 16 address lines are arranged along the top in descending order of mathematical significance. That is, A15 represents 2**15, A14 represents 2**14, etc. Beneath these address lines are the binary representations of each of six hexadecimal addresses. That is, if the processor is pointing to address $0FFF, the various address lines are at a logic 1 or logic 0 level, as shown in the figure. Similarly, the binary representations of the other addresses are shown. Note that if A11 were the highest-order address line available, the processr could reach only from $000 to $FFF, or a total of $1000 (4096 decimal) unique locations. To completely address 4K-byte memory devices, such as 2732 or 2532 EPROMs, those memory devices must have 12 address lines.

If you want to read data from more than one 4K-byte EPROM, you must have additional address lines to divide the memory area into as many *different* blocks of 4096 addresses as you have EPROMs. If you do not divide the memory this way, more than one EPROM will "anwer" each time you try to read memory. Of course, if each EPROM has exactly the same contents, each one will return the same data and there is no problem. Since that is unlikely, you might find that one EPROM is trying to output 10011100 and another 00011111. The output circuit in each EPROM is fighting with the others, and the processor is trying to read digital trash! This situation is known as *bus contention*, and you can have contention at different times during the microprocessor operating cycles. A requisite of computer interfacing is to eliminate any possibility of bus contention so you can predict what will happen at any time during computer operation.

	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0FFF	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1
1000	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1FFF	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1
2000	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
3FFF	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4000	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A000	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

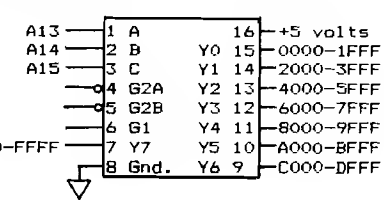
***Figure 1.* A bit map of memory address lines showing which bits are on (logic one) when the microprocessor is running. Note that when an address field fills up (for example, address $FFF), a higher-order address bit must be availsble to designate a larger address. In this case, the next address after $FFF is $1000, which turns on address line A12 for the first time.**

To eliminate bus contention you need to enable only one block of memory at a time, and a decoder is designed to do just that. Figure 2 shows the pinout for one popular decoder — the 74LS138. For those of you unfamiliar with part numbers, the series of IC part numbers beginning with 74 is a logic family called TTL (Transistor-Transistor Logic). This logic is about ten times faster than most microprocessors. TTL parts with LS in the number are lower power parts and are typically used as support devices for microprocessor systems.

The 74LS138 is a moderately complex IC and its operation is defined by the *truth table* shown in figure 3. A truth table defines what outputs result from certain input conditions, and this information allows logic designers to understand how to use the device. Refer back to figure 2, noting certain input and output pin *signatures* (names), which also appear in the chart of figure 3. G1, G2, A, B, and C are all inputs, and all "Y"-named lines are outputs. Note also that the IC has two inputs prefixed with G2 — G2A and G2B. Both these lines are active low (denoted by the circle at the input in figure 2), which means that the lines have to be low for the device to operate. So, in figure 3, if *either* G2A or G2B are high (logic one), the input is disabled. Input G1 is active high (no circle), and so the decoder is disabled when G1 is low. One other common convention is used in figure 3: an "x" means "don't care."

Now examine figure 3 and interpret how a 74LS138 decoder works. In the first line G2 is shown high (that means *either* G2 line), then the device is disabled, and so all four other inputs are "don't care" since they cannot affect a disabled device. When the decoder is disabled, all outputs are high, or inactive. Similarly, in line 2 G1 is shown low, and so all other inputs are don't care and all outputs are high. In the remaining lines, G1 is high and *both* G2 lines are low, and so the decoder is enabled. In the enabled state, each of eight possible combinations of high and low on inputs A, B, and C results in a different *single* output line being low. In other words, changing input levels on inputs A, B, and C create eight unique signals that can be used to select different memory blocks and prevent memory bus contention. You might note one other item with regard to decoders: almost universally, memory devices are selected with ac-

***Figure 2.* The pin-out for one popular decoder, the 74LS138. See text for explanstion of how the decoder operates.**



tive low signals, and so almost all decoders have active low outputs.

Since a decoder responds to memory bus signals and then controls access to memory devices, such operation can be referred to as being "memory mapped;" i.e., part of the memory space. In my first article I referred to a class of I/O (input/output) devices known as a PIA (Programmable Interface Adapter). PIAs reside directly on the processor bus and are selected and controlled by memory bus signals: they are called "memory mapped I/O" devices. A typical PIA is the MC6821 by Motorola. The Color Computer has two PIAs; one reads the keyboard, and one handles all other CoCo hardware — joysticks, cassette recorder interface, serial port, and the D/A (digital/analog) converter that synthesizes the sound tones. Since some of the interfacing experiments will be driven by these PIAs, you should examine the PIA and learn how to program it.

Figure 4 shows the pinout of the 6821 PIA. Note that there are 16 port lines (PA0-PA7; PB0-PB7), 8 data lines (D0-D7), plus RS0, RS1, CS0, CS1, CS2, and R/W*. The RESET* line initializes the PIA during system startup (other lines will be discussed later). Each of the 16 port lines can be set up under program control as either input or output lines by setting a bit in a special register on the PIA. The three CS lines are chip select controls, which are usually driven by address decoders. The two RS (register select) lines are almost always driven by processor address lines, usually A0 and A1.

A 6821 PIA has six registers to control the entire operation of the device. Normally six registers would require three address lines so that each register could have a unique memory address. However, a simple trick allows six registers to be addressed with only two address lines (RS0 and RS1). The internal registers are allocated this way: each of the two 8-bit ports has three registers to control it. The Peripheral Register stores output data that drive the eight package pins associated with the port when the port is acting as an output port; or, if the port is an input port, the Peripheral Register stores in-

(Continued on next page)

***Figure 3.* A truth table explains how a complex logic IC works; this truth table is for the 74LS138 decoder.**

G1	G2	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
x	H	x	x	x	H	H	H	H	H	H	H	H
L	x	x	x	x	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L

Vss	1	40	CA1
PA0	2	39	CA2
PA1	3	38	IRQA*
PA2	4	37	IRQB*
PA3	5	36	RS0
PA4	6	35	RS1
PA5	7	34	RESET*
PA6	8	33	D0
PA7	9	32	D1
PB0	10	31	D2
PB1	11	30	D3
PB2	12	29	D4
PB3	13	28	D5
PB4	14	27	D6
PB5	15	26	D7
PB6	16	25	E
PB7	17	24	CS1
CB1	18	23	CS2*
CB2	19	22	CS0
Vcc	20	21	R/W*

***Figure 4.* Pinout and register addressing scheme for the Motorola MC6821 Programmable Interface Adapter. Note that each output port shares an address with its Data Direction Register, and that Control Register Bit 2 controls which register is addressed. See text for further explanation.**

RS1	RS0	CRA2	Location Selected
0	0	1	Output Port A
0	0	0	Data Direction Register A
0	1	x	Control Register A

RS1	RS0	CRB2	Location Selected
1	0	1	Output Port B
1	0	0	Data Direction Register B
1	1	x	Control Register B

put data to be read back into the processor. The Data Direction Register has one bit for each port bit. If any DDR bit contains a logic one, the corresponding port pin will be an output. Otherwise, with a logic zero in the DDR, the corresponding pin will be an input.

The addressing scheme that selects six registers with only two address bits works this way: the Control Register sets aside bit CR2 (corresponding to the processor D2 bit) as a flag. Under normal operation this flag is set to logic one, and reading or writing the other memory address transfers data to and from the I/O port. If the flag bit is a logic zero, then the other address reaches the Data Direction Register.

Under normal system startup, the RESET* line is connected to the computer's master reset line, and a reset enters a logic zero into each of the six PIA registers. Since the DDR has all zeros, all 16 port lines automatically are set up as inputs. The eight PA lines have internal pull-up resistors, and so these lines go to a logic one. The eight PB port lines switch to a high impedance state; they can drift to any level unless they have an external pull-up or pull-down resistor on them. If no external signal is pulling on the Port A lines, a READ of Port A gives $FF. Without external resistors, a READ of the B Port is indeterminate. If the DDR is written with all ones ($FF), all the port lines immediately pull to logic zero, since the RESET left all zeros in the Peripheral Registers.

To make a controlled startup on the ports of a PIA, the following procedure should be followed to avoid surprises. First, $04 (set bit 2 high) should be written to the Control Register to address the Peripheral Register, then the required initial output data should be written to the Peripheral Registers. Since the RESET left these lines set to input, nothing happens outside the device. Next, write $00 to the Control Register to address the DDR and set logic one for each output pin required. Immediately, the output pins go to the initial values. Finally, write $04 to the Control Register to restore normal configuration. I will deal with PIA programming in more detail later.

MICRO

Reviews in Brief

Product Name: **C64-Link**
Equip. req'd: Commodore 64
Price: $185 (Canadian)
Manufacturer: Richvale Telecommunications
10610 Bayview Avenue #18
Richmond Hill, Ontario L4C 3N8
Canada

Description: *C64-Link* is a module containing circuitry that provides the C64 with IEEE-488 bus compatibility, BASIC 4 commands, a machine-language monitor, and communications routines. It plugs into the 64's cartridge connector, and includes an edge-card connection (like the PET) for a PET-to-IEEE cable. Two programs are included on cassette. One moves the addresss of *C64-Link*'s ROM from $9000-$9FFF to $C000-$CFFF, freeing more RAM for BASIC. The other copies the C64's BASIC ROMs into RAM and replaces the standard serial I/O routines with IEEE ones. No extra RAM is used, but BASIC 4 and the monitor are not available in this configuration.

Pluses: One package adds several desired C64 enhancements. Unit design is sturdy and clean. Software allows great flexibility.

Minuses: Module hangs out from back of C64 without any support. An accident may result in damage to the C64 or *C64-Link*. A new design will include supporting rubber feet.

Documentation: Manual includes summary of capabilities, description of provided software, detailed hook-up instructions for different equipment combinations, and documentation of BASIC 4 and monitor commands.

Skill level required: Beginner

Reviewer: Loren Wright

Product Name: **Star-DOS**
Equip. req'd: TRS-80C Color Computer with disk and 16K memory
Price: $49.90
Manufacturer: Star-Kits
P.O. Box 209
Mt. Kisco, NY 10549

Description: *Star-DOS* is a high-quality disk operating system for the Color Computer that is compatible with Radio Shack Disk BASIC. It features six memory-resident commands and three disk-resident commands. While this is a relatively slim menu, the most commonly needed commands are available. Also, the structure of *Star-DOS* is such that special commands can be added easily by the experienced programmer. Unlike the Radio Shack DOS, *Star-DOS* has 18 user-accessible functions that do most of the I/O needed to support assembly-language programs. For example, the programmer has available routines to read

the keyboard, send characters to the screen, print strings to the screen, etc. The programmer need only develop the central core of his program, with a probable time saving of 50% or more. Also, several routines support disk operations, making it easy to build a custom system that does exactly what the owner requires.

Pluses: *Star-DOS* is inexpensive for a disk program, and is comfortable to use. It is also the only DOS that will run on either the 16K or 32K Color Computer. R/S BASIC compatibility means that a user need not buy a BASIC to have a higher-level language available, and he need not give up the refinements of R/S BASIC that support the special Color Computer hardware and its graphics.

Minuses: *Star-DOS* is new enough that it does not have a large stable of software that will run with it, but this is being remedied. The chief lack is an assembler. An editor/text processor/mailing list/mailing label package is available now.

Skill level required: This product is ideal for the serious disk user who works mainly in assembly language (users who work only with BASIC have no need for any DOS). At the same time the diligent computer user will be able to learn disk system principles and techniques easily.

Documentation: An extremely well-written 55-page manual is furnished. The instructions are thorough and understandable, and a liberal use of examples enhances the learning process. Instructions are included for modifying FLEX-based programs to run under *Star-DOS* when those programs can be made compatible with the stock Color Computer architecture.

Reviewer: Ralph Tenny

Product Name:	**VICMODEM - Model 1600**
Equip. req'd:	VIC-20 (5K or more)
Price:	$109.95
Manufacturer:	Commodore Business Machines, Inc. 487 Devon Park Drive Wayne, PA 19087

Description: The *VICMODEM* package lets the VIC owner join the telecommunications world. A small cartridge-like unit plugs into the VIC-20's user port and enables the VIC to communicate with other computers over telephone lines. The *VICMODEM* connects directly to the telephone *via* the plug that attaches to the handset; no accoustic coupler is required. There is a carrier detect light. The modem has both answer and originate modes to communicate with another VIC or to a time-sharing service like The Source or CompuServe. The package includes a tape with VICTERM, a comprehensive machine-language communications program. Using the menu-driven options

(Continued on next page)

Circle No. 76

Reviews in Brief *(continued)*

in VICTERM, it is easy to set baud rate (up to 300 baud), duplex, word length (seven or eight bits), stop bits, parity, line feed "signals", CBM "half" ASCII or true ASCII, and screen color combinations. VICTERM can avoid having words split from one line to the next. VICTERM also redefines the VIC's function keys to the most frequently used communications codes; e.g., F1 is Control-C, etc. A free subscription to CompuServe is included, with the first hour paid by Commodore.

Pluses: The modem and the software are easy to use and reliable. An outstanding value.

Minuses: The current VICTERM software will not support a disk or a printer, nor is there any way to use the package to transmit or receive a program. The manual refers to a new terminal software package called VICTERM-40 that is being developed and should solve these shortcomings as well as provide an optional 40-character terminal display line.

Documentation: The 20-page booklet is well written and comprehensive.

Skill level required: No special skills.

Reviewer: David Malmberg

MICRO

Circle No. 77

MICRObits

Deadline for MICRObits: 20th of second month before publication; i.e., April 20th for June issue. Send typewritten copy (40-word limit) with $25.00 per insertion. (Subscribers: first ad at $10.00.)

Enhanced OS65D 3.3 C1P-MF

Many new functions such as system commands for catalog control, 10 active files, end of file/end of volume processing, background printing, file append, dynamic file buffers/sectors command file processing, long string read command. Many more! $30. Write for details.

Ray Lydon
20 Eastwood Dr.
Grafton, OH 44044

Lessons in Algebra

An easy and fun way to learn the basic elements of high school algebra. Apple computer diskette $29.95. 30-day money-back guarantee if not satisfied.

George Earl
1302 So. General McMullen Dr.
San Antonio, TX 78237

The State of the Art in Astro-Software

Wide range of astrological and astronomical software of the highest quality. From powerful (and income-producing) astrological charting service packages and printing interpretation packages, to super-accurate computer ephemerises. For all Commodore computers, Apple II Plus, and TRS-80.

Matrix Software
315 Marion Avenue
Big Rapids, MI 49307

Apple II Interfacing

FLY BOARD is a programmable interface that includes a 6522 VIA, 2K bytes of RAM, two 36-inch DIP jumpers, and documentation that makes interfacing easy and fun. Only $129.95 from:

SNAVE SYSTEMS
P.O. Box 957
Niles, IL 60648
(312) 966-4505

Go

Plays at 9 kyu level, scores automatically, can vary board size, etc. $29.00.

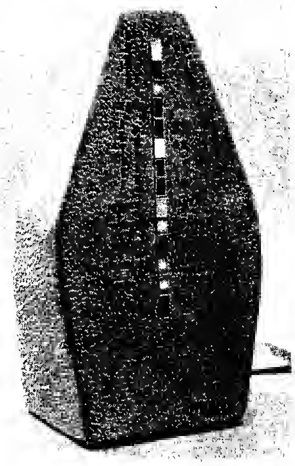
John F. Moore
1145 Alameda #1
Belmont CA 94002

MICRO

MICRO™
Hardware Catalog

Apple Drive Signal Monitor

A unique programming aid for Apple computers monitors drive line status and computer power. The



Drive Signal monitor from Teaco, Inc.

ADASI units provide LED status indication of all lines connecting the floppy drive to the computer. On system bootup, the display shows the activity with a display of flashing lights. First, it indicates the power supply voltages to assure safe operation. Then the multi-colored display shows the status of all lines, for system analysis. ADASI daisy chains between the computer and drive or can be used with the computer alone.

Three are available: The ADASI I (**$59.50**), designed for internal drive connection, and the ADASI II (**$139.50**) and ADASI III (**149.50**), for their respective drives with external connection.

Contact computer stores or Teaco, Inc., P.O. Box E, 2117 Ohio Street, Michigan City, IN 46360; (219)874-6234

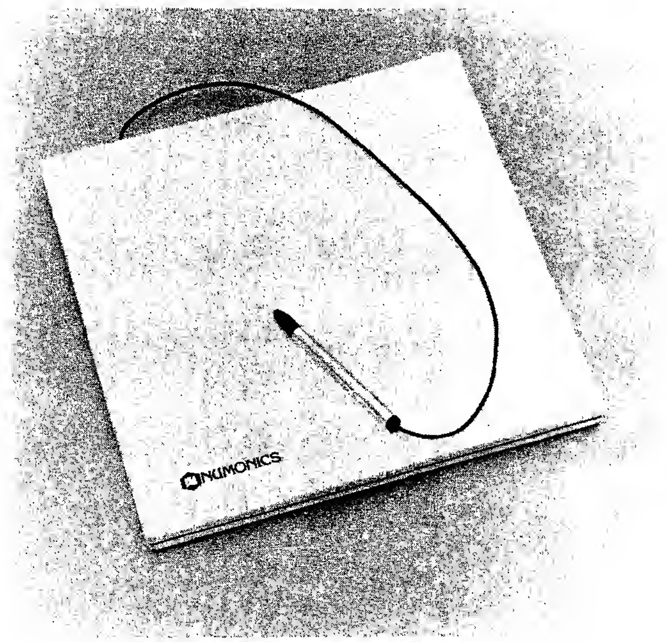
Apple Software Protected

A programmable **Data Lok** for protection of Apple-compatible software allows software to be copied for normal backup. However, the software will run only on machines with Data Lok plugged into the game port socket.

Each series of keys supplied to various software companies is unique to that company. They in turn insert unique code into their software. Quantity pricing is as low as **$12.** The device is available only to

authorized software companies on a controlled basis.

Available from Teaco, Inc., P.O. Box E, Michigan City, IN 46360; (219)874-6234



2200 Series Numonics Pad

A new **electromagnetic digitiaing tablet** features user specification of output data in absolute measurement or the number of user-defined minimum increments down to 0.001" (0.025 mm). Metric or English measurements are also switch-selectable. Multiple interfaces add to its flexibility — dual RS232C, bit parallel or IEEE-488 — and can be output in either the standard serial or packed binary format. It operates in point, stream, incremantal or switch stream modes and measures up to 200 points per second. Other standard features include firmware for self-diagnostics, metric menuing, host override of switch-settable functions, and an inboard audible tone. Optional features include 1, 4, or 16 button cursors, axis rotation and scaling.

An 12"x12" (**$675.00**) and a 20"x20" (**$1275.00**) version are available. Quantity prices on request.

For more information write Numonics, 418 Pierce Street, Lansdale, PA 19446 (215)923-0183

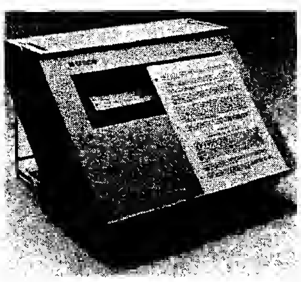
Silence Noisy Printers

Soundtrap quiets printing noise to a level where a business or phone conversation can be conducted standing next to a functioning printer, according to Trace Systems, Inc. with the unit in the upright position, held by the optional stand, the accessory becomes a data holder or copy stand. Soundtrap also

provides storage for paper and simplifies paper feeding and fan folding.

This accessory accomodates most popular printers, including Epson, NEC, Okidata, IBM, and Apple.

For pricing and other information, contact Trace Systems, Inc., 1928 Old Middlefield Way, Mountain View, CA 04043; toll free (800) 24-TRACE, or in CA, call Jim Paige collect at (415) 964-3115



Soundtrap from Trace Systems, Inc.

Commodore Communications

Compack for the Commodore 4032, 8032, 64 is an intelligent terminal communications package that turns the Commodore into a communications control center. It records data to disk, reads data from disk, and sends data to the printer. User programs control the unit to provide remote telemetry, bulletin boards, etc. Price is **$129.95**.

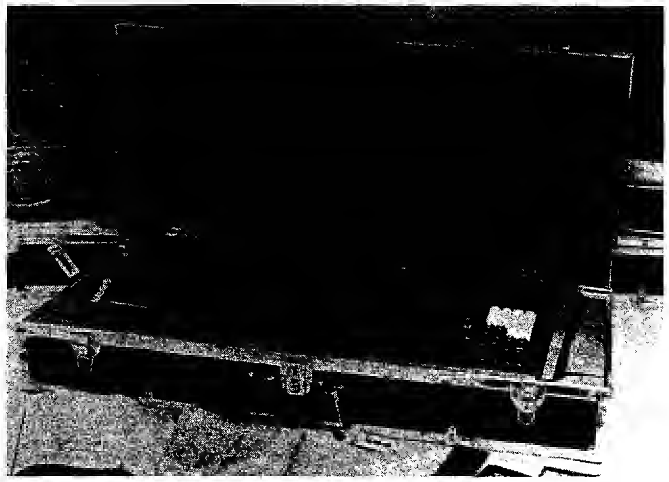
For more information contact CGRS Microtech, P.O. Box 102, Langhorne, PA 19047.

Portable 68000 Trainer

Micro 68000 is a portable 68000 Training/Prototyping System designed for engineers and technicians. It comes with six amp switchig power supply, Versabus 68000 computer board, hexadecimal keyboard, and LED display packaged in a hardwood and dark, plastic case. An optional, padded carrying case is also available.

The 16K byte memory can be any combination of RAM or ROM and includes both Pete-bug keyboard monitor and Tutor-bug providing the user with debug, assembly, disassembly, program entry, and I/O control functions. The expanded display board shows entries in both hexadecimal and binary. The computer board contains two RS-232 ports and 32 bits of parallel I/O. Micro 68000 comes with Lance Leventhals's "68000 Assembly Language Programming," Motorola's "68000 Users Manual," and CSA's "Micro 68000 User's Manual." Price is **$1498.00**

For more information contact Computer Systems Associates, 7562 Trade St., San Diego, CA 92121; (619)566-3911



Micro 68000 from Computer Systems Associates

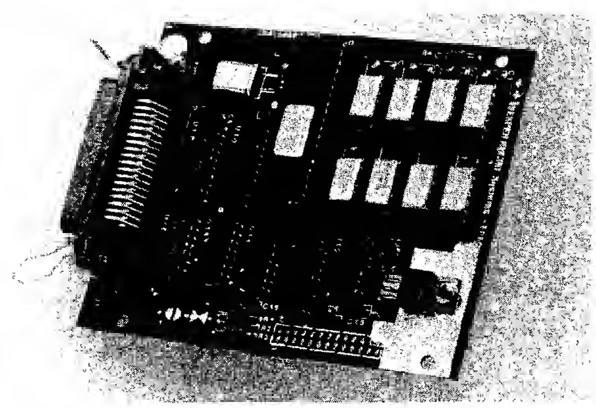
64K Buffer for Epson Printers

A new printer buffer card, **Wizard-EBI Epson Buffered Interface** mounts inside all Epson MX Series Printers. It allows the computer to dump its print data into the buffer quickly freeing up the computer. The printer continues to print at its own rate of speed while the computer goes on to the next task.

The Wizard-EBI does not change the printer characteristics, so no special software or cabling is required. Delivered ready to install by a simple procedure, the buffer is available with 8K (**$139.00**), 16K (**$158.00**), 32K (**$200.00**), or 64K (**$280.00**) characters, depending on the number of RAM chips plugged into sockets provided on the card. The Wizard-EBI has a Centronics-compatible parallel interface identical to that of the Epson printer.

For more information contact Wesper Microsystems, 3188 Pullman Street, Costa Mesa, CA 92626; (800) 850-8737, or in CA (714) 850-1666.

Wizard-EBI Epson Buffered Interface from Wesper Microsystems



MICRO™

Software Catalog

Dark Crystal for the Apple

The Dark Crystal Adventure Game offers significant advancements in graphics, language, and programming, according to its producers. The player becomes Jen, the hero of the movie "The Dark Crystal," and controls actions by typing commands on the computer keyboard.

The plot of the game concerns a quest for a missing shard (as it did in the motion picture), which must be replaced by Jen in the broken crystal in order to save the world.

Price is **$39.95**. Available from Sierra On-Line Inc., Sierra On-Line Building, Coarsegold, CA 93614; (209) 683-6858.

Elementary Fun

Rhymes and Riddles for the Atari, IBM/PC, and Apple II Plus, contains three-letter guessing games, nursery rhymes, riddles, and famous sayings. In each game you press letters to fill in the blanks and complete the lines. Once you have correctly completed the lines you are rewarded with colorful graphics and sound.

Price is **$29.95.** Available from Spinnaker Software Inc., 215 First St., Cambridge, MA 02142

Help With Math

Elements of Mathematics for the Apple II Plus assists in the instruction of the elements of mathematical functions. Content includes: adding fractions (common denominators); reducing fractions; adding fractions (unlike denominators). Student recordkeeping is provided.

Price is **$90.00**. Available from Electronic Courseware Systems, Inc., P.O. Box 3274, Station A, Champaign, IL 61820; (217) 359-7099.

Stock Market on the Apple

This Stock Market Utility Package, **DOW 2000/ OPTION43/BE. POINT7**, will determine price projections based on a stock's BETA coefficient or Relative Strength # and the Dow Jones Average. Projections are made as you vary the DOW, on one stock or entire portfolio with single scan, quick scan, or variable scan of values. The option program will give you the percent of cost increase over the option months to determine which month and strike price option to buy for a given stock. BE.POINT7 will determine your break-even point for options or securities.

Price is **$23.95** (booklet $4.95 extra). Available from Bit'n Pieces Series, P.O. Box 7035, Erie, PA 16510-0035.

"The Dark Crystal" adventure game recreates scenes from the motion picture.

Improve Your Typing

Typing Package for the VIC-20 consists of three different programs on a single tape and assists typing students. One program, WARMUP, takes the student through a series of finger exercises of increasing difficulty. The other programs give the student drill on longer blocks of text. A score is indicated for all programs. The package is a supplement to a school course or self-teaching text.

Price is **$12.75**. Available from MFJ Electro-Enterprises, P.O. Box 13076, Kanata, Ont. K2K 1X3 Canada; (613) 592-2962.

VIC Adventure

In **Zorlok** an adventure game for the VIC-20, you are the great, great grandson of Zorlok the wizard, and you have inherited a quest! You must enter his castle, wipe out a plague of monsters, and regain his

SCIENTIFIC SOFTWARE ASSOCIATES, LTD.

Q-card

Questionnaire Analysis Software

- Microcomputer based
  Avoid the expense of contract services -- do everything in-house on your own Apple II+ microcomputer
- Easy data entry
  Avoid time consuming keypunching. Uses respondent-marked cards entered with an Optical Mark Reader (keyboard entry also possible).
- Comprehensive data analysis
  Sort on any variable(s), tally all responses, conduct cross tabs, correlations, linear regression, frequency distributions, and more.
- Complete editing capabilities
  Weight items, derive composites, add or delete items, and more.
- Easy-to-use
  Programs are user friendly, menu driven, and interactive. No special computer expertise is required.

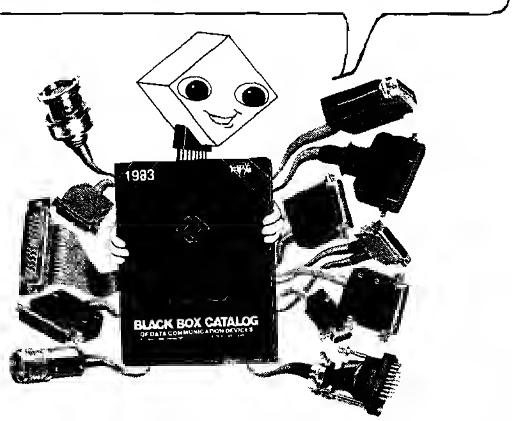
Call or send for more information today.

SCIENTIFIC SOFTWARE ASSOCIATES, LTD.
BOX 208 · WAUSAU, WI. 54401
TELEPHONE: (715) 845-2066

Apple II+ is a registered trademark of Apple Computer, Inc.

Circle No. 49

Boxey Says: 'The place to find the Cable you need is in my Catalog!"



No matter what type of data cable you need, you can find it in the **BLACK BOX® Catalog**. We carry 23 types of cables to fit every popular interface (17 in all). Data Cables are available cut to length and terminated to your specs or in bulk with separate connectors for on-site installation.

The 1983 Edition of the **BLACK BOX®Catalog** contains 282* different data communications products, including cables.

Send for your copy today. It's Free!

*56 models of Data Switches, 14 Test Sets, 7 Modem Eliminators, 6 Line Drivers, 5 Protocol Converters, 9 Communication Adapters, 8 Printer Interfaces, 8 Terminal/Line/Modem sharing devices, Tools, etc., etc.

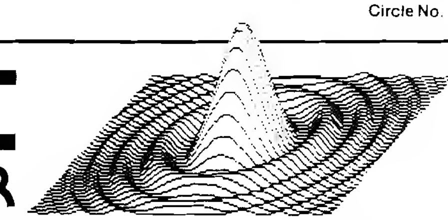
Phone or write:

BLACK BOX® CATALOG
A MICOM COMPANY
Dept. SQ • P.O. Box 12800 • Pittsburgh, PA 15241
(412) 746-2910 • TWX 510-697-3125

Circle No. 69

IMAGE PRINTER][

THE FULL SERVICE PICTURE PRINTER



BEYOND CONVENTIONAL PROGRAMS.

The new, improved IMAGE PRINTER goes beyond the capabilities of conventional picture printing programs for the Apple //. Even owners of graphic printer interface cards will find that IMAGE PRINTER greatly expands their picture printing abilities!

FULL SERVICE MEANS MORE FEATURES.

IMAGE PRINTER starts you out on the fast track by helping you "capture" the HI-RES pictures from your favorite programs—even the copy-protected ones!

IMAGE PRINTER then lets you customize the picture the way you want by adding titles, lines, boxes, color filling portions of the screen, or even scrolling the entire picture in any of four directions.

After you polish your picture, IMAGE PRINTER lets you easily select any portion of it to be printed. The resulting image can be shrunk or expanded and then printed either vertically or horizontally, anywhere on the printed page.

FEATURES	IMAGE PRINTER	GRAPHIC PRINTER CARDS	OTHER PICTURE PROGRAMS
Capture pictures from programs	X	—	—
Save pictures on diskette	X	—	X
Menu driven for ease of use	X	—	X
Unprotected, modifiable	X	—	—
Add titles, borders, lines & boxes	X	—	—
Color fill portions of picture	X	—	—
Scroll pictures 4 ways	X	—	—
View picture before printing	X	—	X
Print any portion of picture	X	—	X
Select a portion of a picture using a graphic "window"	X	—	—
Print ½ to 6x normal size	X	—	—
Print horizontally or vertically	X	X	X
Print anywhere on page	X	—	X

Image Printer works with over 30 different printers and 20 different interface cards.

DON'T SETTLE FOR LESS!
GET THE FULL SERVICE PICTURE PRINTER! $49.95

Sensible Software, Inc. 6619 Perham Drive
West Bloomfield, Michigan
48033 (313) 399-8877

Visa and Mastercard Welcome *Add 1.25 postage and handling per program*

Circle No. 71

Software Catalog *(continued)*

treasures. Multiple skill levels are provided.

Price is **$39.95**. Available from MicRo Information Systems, P.O. Box 73, Wayne, NJ 07470; (201) 696-3296.

FILEX for PET/CBM

FILEX for PET/CBM allows you to read and write IBM "BASIC Data Exchange" diskettes. The system allows information exchange between mainframe/mini computers, and remote CBM machines. Requires 32K, PEDisk 8" floppy.

Price is **$245.00**, which includes ROM and manual. Available from CGRS Microtech, P.O. Box 102, Langhorne, PA 19047; (215) 757-0284.

Light Typing

MasterType™, for the Atari 400 or 800 and Apple II provides typing instruction in game format. You must zap the enemy word by typing it correctly or the word will zap you. Eighteen lessons graduate from home letter recognition to eight-letter words, numbers, and BASIC programming words. You can create your own lesson to meet individual needs.

Price is **$39.95**. Available from Lightning Software, Inc., P.O. Box 11725, Palo Alto, CA 94306.

SXR Plus for the Apple II

SXR Plus produces a sorted cross reference of an Applesoft source program. Variables are always included and the user has the option to include/exclude referenced line numbers, numeric constants and or strings. All information is presented in a single alphabetized list; the user can select either a 40- or an 80-column format. A search feature is also included.

Price is **$39.95.** Available from Prasek Computer Systems, Inc., P.O. Box 2427, Santa Clara, CA 95055; (408) 554-0420, or computer stores.

Geography on the Color Computer

Geography Pac for the TRS-80 Color Computer, an educational program, helps you learn world or U.S. geography in an enjoyable way. You need 16K Extended BASIC machine language and a cassette tape recorder.

The **$29.95** price includes cassette tape of U.S., Europe, Asia, Africa, and South Central America.

Available from Spectral Associates, 141 Harvard Avenue, Tacoma, WA 98466.

Apple II Game

Monster Mash is an arcade-style game for the Apple II and Apple III (in emulation mode) computers. It's your job to keep the rowdy monsters in the graveyard, and all you have to do it with is your new Monster Masher system and quick reflexes. The game offers many different skill levels and control configurations. 48K required.

Price is **$29.95.** Available from The Software Farm, 3901 So. Elkhart St., Aurora, CO, 80014; (303) 690-7559.

Commodore Word Processor

Copy-Writer for the Commodore PET/CBM and Commodore 64 is a second-generation word processor containing features of the best word processing systems; pagination, numbering, justification, spacing, searching. block moves, etc. It also contains capabilities for double column, shorthand, and graphics. Peridodic updates are included.

Price is **$185.00**.Available from CGRS Microtech, P.O. Box 102, Langhorne, PA 19047; (215) 757-0248

MICRO

Circle No. 12

BULLETIN BOARD

FORUM-80 Augusta, GA	(803) 279 5392	
FORUM-80 Charleston, SC	(803) 552 1612	*24
FORUM-80 Cleveland, OH	&(216) 486 4176	
FORUM-80 #2, Denver, CO	(303) 399 8858	*24
FORUM-80 El Paso, TX	(915) 755 1000	*24
FORUM-80 Family Historian Fairfax, VA	(703) 978 7561	
FORUM-80 Ft. Lauderdale, FL	(305) 772 4444	*24
FORUM-80 Hull, England	(011) 44 482 859169	
FORUM-80 Kansas City, MO	#1 &(816) 861 7040	*24
FORUM-80 Kansas City, MO	&(816) 931 9316	
FORUM-80 Las Vegas, NV	(702) 362 3609	*24
FORUM-80 Linden, NJ	(201) 486 2956	*24
FORUM-80 Medford, OR	(503) 535 6883	*24
FORUM-80 Medical, Memphis, TN	(901) 276 8196	*24
FORUM-80 Monmouth, Brielle, NJ	(201) 528 6623	*24
FORUM-80 Montgomery, AL	(205) 272 5069	
FORUM-80 Nashua, NH	(603) 882 5041	
FORUM-80 Prince William County, VA	(703) 670 5881	*24
FORUM-80 San Antonio, TX	!(512) 655 8143	
FORUM-80 Seattle, WA	(206) 723 3282	
FORUM-80 Sierra Vista, AZ	(602) 458 3850	*24
FORUM-80 Shreveport, LA	(318) 631 7107	*24
FORUM-80 Westford, MA	(617) 692 3973	
FORUM-80 Wichita, KA	&(316) 682 2113	*24
FORUM-80 Wichita Falls, TX	(817) 855 3916	
FORUM-80 Wild goose board, Tampa, FL	(813) 988 7400	
Greene Machine, WPB, FL	(305) 965 4388	-so
Greene Machine Fricaseed Chicken, Arcadia, CA	(213) 445 3591	*24
Greene Machine, Riverside, CA	!(714) 354 8004	
Greene Machine Corsair, WPB, FL	!(305) 968 8653	
Greene Machine, Los Alamitos, CA	!(213) 431 1443	
Greene Machine, Rome, NY	!(315) 337 7720	
Greene Machine, Irvine, CA	!(714) 551 4336	
Greene Machine, Temple City, CA	!(213) 287 1363	
HBBS Denver, CO	(303) 343 8401	*24
HBBS El Paso, TX	(915) 592 1910	
HBBS Oklahoma City, OK	(405) 848 9329	*24
MCMS C.A.M.S. Chicago, IL	#1&(312) 927 1020	*24
MCMS J.A.M.S. Lockport, IL	(815) 838 1020	*24
MCMS L.A.M.S. Round Lake, IL	(312) 740 9128	
MCMS P.C.M.S. Wheaton, IL	!&(312) 462 7560	*24
MCMS Metro West Database, Chicago, IL	&(312) 260 0640	*24
MCMS NC Software, Minneapolis, MN	(612) 533 1957	*24
MCMS WACO Hot Line, Schaumburg, IL ...... pvt	(312) 351 4374	*24
NET-WORKS ABC, Kansas City, MO	(816) 483 2526	
NET-WORKS Apple Grove, Dallas, TX	(214) 644 5197	
NET-WORKS Apple Shack, Dallas, TX	(214) 644 4781	*24
NET-WORKS Armadillo, Grand Forks, ND	(701) 746 4959	
NET-WORKS Beach BBS, Pensacola, FL	(904) 932 8271	
NET-WORKS Big Apple, Miami, FL	(305) 948 8000	
NET WORKS C.A.M.S., Decatur, IL	(217) 429 5541	
NET-WORKS Charleston, WV	(304) 345 8280	
NET-WORKS Chipmunk, Hinsdale, IL	(312) 323 3741	*24
NET-WORKS Coin Games, Los Angeles, CA	(213) 336 5535	
NET-WORKS COMM Center NW3N AGAD, Laurel, MD	(301) 953 3341 (301) 792 0305	
NET-WORKS Computer City, Providence, RI	(401) 331 8450	*24
NET-WORKS Computer Emporium, Des Moines, IA	(515) 279 8863	
NET-WORKS Computer Emporium, San Jose, CA	(408) 227 0227	
NET-WORKS Computer Market, Honolulu, HI	(808) 521 7312	
NET-WORKS Computer Pro, Ft. Worth, TX	(817) 732 1787	
NET-WORKS Computer Station, St. Louis, MO	(314) 432 7120	
NET-WORKS Computer Store, Honolulu, HI	(808) 488 7756	
NET-WORKS Computer World, Los Angeles, CA	(213) 859 0894	*24
NET-WORKS Crescent City, Baton Rouge, LA	(504) 454 6688	
NET-WORKS Dallas, TX	(214) 361 1386	*24
NET-WORKS Dayton, OH	(513) 223 3672	
NET-WORKS Eclectic Computer Sys., Dallas, TX	(214) 239 5842	
NET-WORKS Granite City, IL	(618) 877 2904	
NET-WORKS Greenfield, IN	(317) 326 3833	*24
NET-WORKS Hacker-net, Dallas, TX	(214) 824 7160	
NET-WORKS Hawaii	(808) 521 7312	
NET-WORKS Hawaii Connection, Honolulu, HI	(808) 423 1593	*24
NET-WORKS MAGIE, Galesburg, IL	(309) 342 7178	
NET-WORKS Magnetic Fantasies, Los Angeles, CA	(213) 388 5198	
NET-WORKS MICRO-BBS Chelmsford, MA	(617) 256 1446	
NET-WORKS Missouri	(314) 781 1308	
NET-WORKS New York, NY	(212) 410 0949	
NET-WORKS North Parks, Chicago, IL	(312) 745 0924	
NET-WORKS Pirate's Harbor, Boston, MA	(617) 720 3600	
NET-WORKS Pirate's Lodge ???	(914) 634 1268	
NET-WORKS Pirate's Ship, Chicago, IL	(312) 935 2933	*24
NET-WORKS Pirate's Trek ???	(516) 627 9048	
NET-WORKS Portsmouth, NH	(603) 436 3461	
NET-WORKS Softworx, West Los Angeles, CA	(213) 473 2754	
NET-WORKS Sparklin' City, Corpus Christi, TX	(512) 882 6569	
NET-WORKS Toronto, Ontario, CN	(416) 445 6696	*24
NET-WORKS Warlock's Castle, St. Louis, MO	(618) 345 6638	
NET-WORKS Winesap, Dallas, TX	(214) 824 7455	
NET-WORKS ???	(914) 725 4060	
ONLINE CDC, San Diego, CA	(619) 452 6011	
ONLINE Computerland, Montreal, Quebec, CN	(514) 931 0458	*24
ONLINE Dickinson's Movie Guide, Mission, KS	(913) 432 5544	
ONLINE Indianapolis,IN.ID# = GUES, pswd = pass	(317) 787 9881	*24
ONLINE Saba, San Diego, CA	(619) 692 1961	*24
ONLINE Santee, CA....ID # = GUEST, pswd = PASS	(619) 561 7271	*24
PASBBS Torrance, CA	#1 (213) 516 7089	*24
PBBS Co-operative Comp Svc, Palatine, IL	(312) 359 9450	*24
PET BBS S.E.W.P.U.G., Racine, WI	(414) 554 9520	*24
PET BBS Commodore Comm., Lake St. Louis, MO	(314) 625 4576	*24
PMS -**IF**, Anaheim, CA	(714) 772 8868	*24
PMS -Anchorage, AK	(907) 344 8558	
PMS -Apple Bits, Kansas City, MO	(913) 3[illegible]1 3502	*24
PMS -Apple Guild, Weymouth, MA	(617) 767 1303	*24
PMS -Baltimore, MD	(301) 764 1995	*24

PMS -Campbell, CA	(408) 370 0873	*24
PMS -Century 23, Las Vegas, NV	(702) 878 9106	*24
PMS -Chicago, IL	(312) 373 8057	*24
PMS -Cincinnati, OH	(513) 671 2753	
PMS -Computer City, Danvers, MA	(617) 774 7516	
PMS -Computer Merchant, San Diego, CA	(619) 582 9557	
PMS -Computer Solutions, Eugene, OR	(503) 689 2655	*24
PMS -Datel Systems Inc., San Diego, CA	!(619) 271 8613	*24
PMS -Downers Grove/SRT, Downers Grove, IL	(312) 964 6513	
PMS -El Cajon, CA	(619) 579 0553	
PMS -Ellicott City, MD	(301) 465 3176	
PMS -Escondido, CA	(619) 746 0667	
PMS -Ft. Smith Comp. Club, Ft. Smith, AK	(501) 646 0197	
PMS -Gulfcoast, Freeport, TX	(713) 233 7943	*24
PMS -Indianapolis, IN	(317) 787 5486	*24
PMS -Lakeside, CA. (type PMS to activate)	(619) 561 7271	*24
PMS -Los Angeles, CA	(213) 334 7614	*24
PMS -Massillon, OH	(216) 832 8392	*24
PMS -McGraw Hill Books, New York, NY	(212) 997 2488	
PMS -Minneapolis, MN	(612) 929 6699	*24
PMS -I.A.C., Lake Forest, IL	(312) 295 6926	*24
PMS -O.A.C., Woodland Hills, CA	(213) 346 1849	*24
PMS -Pikesville, MD	(301) 653 3413	
PMS -Pleasanton, CA	(415) 462 7419	*24
PMS -Portland, OR	!(503) 245 2536	
PMS -Portola Valley, CA	(415) 851 3453	*24
PMS -RAUG, Akron, OH	(216) 867 746?	*24
PMS -Rutgers Univ. Microlab, Piscataway, NJ	(201) 932 3887	
PMS -Santa Cruz, Aptos, CA	(408) 688 9629	*24
PMS -Santee, CA	#1 (619) 561 7277	*24
PMS -Shrewsbury, NJ	(201) 747 6768	
PMS -Software Unltd, Kenmore, WA	(206) 486 2368	*24
PMS -Twin Cities, Minneapolis, MN	!(612) 929 8966	
PMS -Your Computer Connection, KS Cty, MO	!(913) 677 1299	
PSBBS Baltimore, MD	(301) 994 0399	*24
PSBBS Washington, DC	(202) 337 4694	*24
RATS Systems	#1 (201) 887 8874	
RATS Homewood, IL	(312) 957 3924	
RATS Wenonah, NJ	(609) 468 5293	
RATS Wenonah, NJ #2	(609) 468 3844	
RCP/M A.B. Dick Co., Niles, IL	&(312) 647 7636	*24
RCP/M AIMS Hinsdale, IL	(312) 789 0499	*24
RCP/M Arlington, VA	(703) 536 3769	
RCP/M CBBS CP/M Net Simi Valley, CA	(805) 527 9321	
RCP/M CBBS Columbus, OH	(614) 272 2227	*24
RCP/M CBBS Dallas, TX	(214) 931 8274	
RCP/M CBBS Frog Hollow, Vancouver, BC, CN	(604) 873 4007	*24
RCP/M CBBS Pasadena, CA	(213) 799 1632	*24
RCP/M CBBS RLP, MacLean, VA	(703) 524 2549	*24
RCP/M CBBS Sacramento, CA	(916) 483 8718	*24
RCP/M Chuck Forsberg, OR	(503) 621 3193	
RCP/M Collossal Oxgate, San Jose, CA	(408) 263 2588	
RCP/M CUG-NOTE, Denver, CO	(303) 781 4937	*24
RCP/M CUG NODE, PA State College	(814) 238 4857	*24
RCP/M Detroit, MI	(313) 584 1044	-rb
RCP/M Geneseo, IL	(309) 944 5455	
RCP/M HAPN Hamilton, Ontario, CN	(416) 335 6620	*24
RCP/M IBM PC, Niles, IL	(312) 259 8086	
RCP/M Logan Square, Chicago, IL	(312) 252 2136	
RCP/M MCBBS Keith Petersen, Royal Oak, MI	(313) 759 6569	-rb
RCP/M MCBBS Ken Stritzel, Flanders, NJ	(201) 584 922?	*24
RCP/M MCBBS Superbrain, Lexington, MA	$&(617) 862 0781	*24
RCP/M MCBBS TCBBS Dearborn, MI	(313) 846 6127	*24
RCP/M Mississauga HUG, Toronto, Ont., CN	$&(416) 826 5394	*24
RCP/M NEI, Chicago, IL	(312) 949 6189	
RCP/M Palatine, IL	&(312) 359 8080	*24
RCP/M RBBS Allentown, PA	(215) 398 3937	*24
RCP/M RBBS ANAHUG, Anaheim, CA	(714) 774 7860	*24
RCP/M RBBS Arvada Elect., Colorado Springs, CO	(303) 634 1158	*24
RCP/M RBBS BBS Valley	!(213) 360 5053	
RCP/M RBBS Boulder, CO	(303) 499 9169	
RCP/M RBBS Bethesda, MD	(301) 229 3196	
RCP/M RBBS Brewster, NY	(914) 279 5693	
RCP/M RBBS Comp. Tech. Assoc., El Paso, TX	(915) 533 2202	*24
RCP/M RBBS Computerized Services, Tampa, FL	(813) 988 7400	*24
RCP/M RBBS Computron, Edmonton, Alberta, CN	(403) 482 6854	*24
RCP/M RBBS Cranford, NJ	(201) 272 1874	
RCP/M RBBS DataTech 001, San Carlos, CA	#1$&(415) 595 0541	*24
RCP/M RBBS DataTech 004, Sunnyvale, CA	(408) 732 2433	
RCP/M RBBS DataTech 006, San Francisco, CA	(415) 563 4953	
RCP/M RBBS Edmonton, Alberta, CN	&(403) 454 6093	*24
RCP/M RBBS El Paso, TX	(915) 598 1668	
RCP/M RBBS Fort Mill, SC	(803) 548 0900	*24
RCP/M RBBS GFRN Dta Exch. Garden Grove, CA	$&(714) 534 1547	*24
RCP/M RBBS GFRN Dta Exch. Palos Verdes, CA	$&(213) 541 2503	*24
RCP/M RBBS Grafton, VA	(804) 898 7493	
RCP/M RBBS Houston, TX	(713) 497 5433	
RCP/M RBBS Huntsville, AL	(205) 895 6749	-rb
RCP/M RBBS Laurel, MD	(301) 953 3753	*24
RCP/M RBBS Larkspur, CA	(415) 461 7726	*24
RCP/M RBBS Marin County, CA	(415) 383 0473	*24
RCP/M RBBS Mike's, Milwaukee, WI	!(414) 647 0903	
RCP/M RBBS MUG, Mission, KS	&(913) 362 9583	*24
RCP/M RBBS Napa Valley, CA	(707) 253 1523	
RCP/M RBBS Ocean, NJ	&(201) 775 8705	
RCP/M RBBS Piconet Oxgate, Mountain View, CA	(415) 965 409?	*24
RCP/M RBBS San Jose Oxgate, San Jose, CA	(408) 287 5901	*24
RCP/M RBBS Surrey, Vancouver, BC, CN	(604) 584 2643	*24
RCP/M RBBS Pontiac, MI	(313) 338 8515	
RCP/M RBBS Paul Bogdanovich, NJ	(201) 747 7301	
RCP/M RBBS Rochester, NY	(716) 223 1100	*24
RCP/M RBBS Rutgers, New Brunswick, NJ	(201) 932 3879	*24
RCP/M RBBS San Diego, CA	$&(619) 273 4354	*24
RCP/M RBBS Sofwaire Store, Los Angeles, CA	(213) 296 5927	*24
RCP/M RBBS Software Tools, Austrailia	(02) 997 1836	
RCP/M RBBS Southfield, MI	(313) 559 5326	*24
RCP/M RBBS Westland, MI	(313) 729 1905	-rb
RCP/M RBBS Woodstock, NY	(914) 679 8734	*24
RCP/M RBBS Yelm, WA	(206) 45[illegible] 3[illegible]	-rb
RCP/M Silicon Valley, CA	(408) 246 5014	*24

BULLETIN BOARD

RCP/M SJBBS Bears ville, NY	(914) 679 6559	-rh
RCP/M SJBBS Johnson City, NY	(607) 797 6416	
RCP/M Terry O'Brien, Vancouver, BC, Canada	(604) 584 2543	
Remote Northstar Atlanta, GA	#1 (404) 926 4318	*24
Remote Northstar Denver, CO	(303) 444 7231	
Remote Northstar Largo, FL	(813) 535 9341	*24
Remote Northstar NASA, Greenbelt, MD	(301) 344 9156	
Remote Northstar Santa Barbara, CA	(805) 682 7876	
Remote Northstar Santa Barbara, CA	(805) 964 4115	
Remote Northstar Virginia Beach, VA	(804) 340 5246	
ST80-CC Lance Micklus, Inc. Burlington, VT	# 1(802) 862 7023	*24
ST80-PBB Monroe Camera Shop, Monroe, NY	(914) 782 7605	
TCBBS B.A.M.S. New York, NY	(212) 362 1040	*24
TCBBS Leigh's Computer World, NY	(212) 879 7698	
TCBBS AstroCom, New York, NY	#1!(212) 799 4649	
TRADE-80 Alhany, GA	(912) 439 7440	*24
TRADE-80 Ft. Lauderdale, FL	#1 (305) 525 1192	
TRADE-80 Omaha, NE	(402) 292 6184	
TRADE-80 Erie, PA	(814) 898 2952	*24

MISCELLANEOUS OR UNKNOWN SYSTEM TYPES

ABBIES Info System Phila, PA	(215) 237 6908	
ABBS (?) Queens, NY	(212) 896 0519	
(?) (Western Massachusettes)	(413) 637 3515	
ACE Oregon	(503) 343 4352	
Adventure BBS	(516) 621 9296	
All Night BBS	(213) 564 7636	
Alpha, Tampa, FL .... acct# = ABCD00, pwd = TRYIT	(813) 251 4095	*24
Aphrodite-E	(201) 790 5910	-so
Apollo's Chariot, Apollo, FL	(813) 645 3669	
Apple-Can Ontario	(416) 781 1796	
Apple-Gram	(313) 295 0783	*24
Applecrackers, Columbus, OH	(614) 475 9791	*24
Apple Tree BBS California	(714) 963 7222	
ARBB Seattle, WA	(206) 546 6239	
Armadillo Media Services, Houston, TX	(713) 444 7098	
Astro BBS New York	(212) 787 5520	
ATBBS Honolulu, HI	(808) 833 2616	
Aviators Bulletin Board, Sacramento, CA	(916) 393 4459	
Bathroom Wall BBS, San Antonio, TX	(512) 655 8143	
Baton Rouge Data System, Baton Rouge, LA	(504) 926 0181	
Blue BOSS IBM PC, Berkeley, CA	(415) 845 9462	*24
BBS Annandale, VA	(703) 978 9754	
BBS Apollo, Phoenix, AZ	!(602) 246 1432	*24
BBS Atlanta, GA	(404) 939 1520	
BBS B.R., Los Angeles, CA	(213) 394 5950	*24
BBS Computer Applications Co., Poland, OH	(216) 757 3711	
BBS Homestead, FL	(305) 246 1111	
BBS Living Videotext, Menlo Park, CA	(415) 327 8876	*24
BBS Metro Detroit, MI	!(313) 455 4227	-so
BBS Pensacola, FL	(904) 477 8783	
BBS SUE Milwaukee, WI	(414) 483 4578	
BBS The BULL, Toronto, Ontario, CN	!(416) 423 3265	-so
BBS-80 DALTRUG, Dallas, TX	(214) 235 8784	*24
Big Top Games System, Milwaukee, WI	(414) 259 9475	
Boston Information Exchange, Boston, MA	&(617) 423 6985	*24
Bronx BBS, NY	(212) 933 9459	
Bradley Computer BBS	(813) 734 7103	
BSBB Tampa, FL	(813) 885 6187	
Call A.P.P.L.E. Washington	(206) 524 0203	
C.A.M.S. Illinois	(217) 429 5505	
Capital City BBS, Albany, NY	(518) 346 3592	*24
Carrier 2 Alexandria, VA	(703) 823 5210	
C-HUG Bulletin Board, Fairfax, VA	(703) 360 3812	*24
COMM-80 Queens, NY	(212) 897 3392	*24
Communique-80 Livingston, NJ	(201) 992 4847	
Compuque-80, Houston, TX	(713) 444 7041	*24
Compusystems, Columbia, ??	(803) 771 0922	
Computer Connection	(213) 657 1799	
Datamate, Canoga Park, CA	#1 (213) 998 7992	-so
Davy Jones Locker	(313) 764 1837	
Dimension-80 Orange, CA	(714) 974 9788	
Distra-Soft, Montreal, Quebec, CN	(S14) 327 S764	*24
Dragon's Game System .... (pass = DRAGON)	(213) 428 5206	
Drummer	(415) 552 7671	-so
Electric Line Connection, Sherman Oaks, CA	(213) 789 9512	
Experimental-80 Kansas City, MO	(913) 676 3613	
FGS/RP Games Illinois	(312) 743 8176	
Heath BBS Lakewood,NJ	(201) 363 3122	
Hermes-80 Allentown, PA	(215) 434 3998	
HEX Silver Spring, MD	%(301) 593 7033	*24
IDBN Info-Net, Costa Mesa, CA	(714) 545 7359	
INFOEX-80 West Palm Beach, FL	(305) 683 6044	*24
INFOEX-80 Akron, OH	!(216) 724 2125	*24
Infoport Ontario	(416) 278 3267	
Info-System Ontario	(416) 622 2462	
Irvine Line, Irvine, CA	(714) 551 4336	
JCTS Redmond, WA	(206) 883 0403	*24
Kinky Kumputer, San Francisco, CA	(415) 626 5465	-so
Kluge Computer	$&(213) 947 8128	*24
L.A. Interchange, Los Angeles, CA	(213) 631 3186	*24
Lehigh Press BB, PA	#1 (215) 435 3388	
Lethbridge Gaming System, Lethbridge, AB	(403) 320 6923	
LITHO/NET	(800) 831 6964	
Long Beach Community Computer	(213) 591 7239	*24
Mail Board-82 Seattle, WA	(206) 527 0897	*24
Micro-80 West Palm Beach, FL	(305) 686 3695	
Micro Design, Houston, TX	(713) 864 4672	
Micro Informer	(813) 884 1506	
Midwest, St. Louis, MO	(314) 227 4312	-so
Mini-Bin Seattle, WA	(206) 762 5141	*24
MOUSE-NET Orlando, FL	(305) 277 0473	*24
MRCBBS	(415) 968 1093	-so
MSG-80 Everett, WA	(206) 334 7394	
NBBS Norfolk, VA	(804) 444 3392	
NESSY Chicago, IL	1 (312) 773 3308	
New England Comp. Soc., Maynard, MA	(617) 897 0346	
New Jersey TELECOM	#1 (201) 635 0705	*24
North Orange County Computer Club	(714) 633 5240	
Novation CO., Los Angeles, CA ...... pass = CAT	(213) 881 6880	
Nybbles-80 Elmsford, NY	(914) 592 5385	
Nybbles-80 NY	(212) 626 0375	
OARCS Portland, Oregon	(503) 641 2798	
OCTUG Orange County, Garden Grove, CA	(714) 530 8226	
Ohio Valley BBS	(614) 423 4422	
Oracle North Hollywood, CA	(213) 980 5643	-so
Orange County Data Exchange, Garden Grove, CA	(714) 537 7933	
OS UNA Scarsdale, NY	(914) 72S 4060	
PACS Online Phila, PA	(215) 342 4013	
PACE -NET Pittsburg, PA	(412) 655 2652	
Personal Msg. System-80, Deerfield Bch, FL	&(305) 427 6300	*24
PET BBS Commodore, Chicago, IL	(312) 397 0871	*24
PET BBS AVC Comline, Indianapolis, IN	(317) 255 5435	*24
PET BBS KCPUG, Kansas City, KS	(816) 356 2382	*24
PET BBS Nortec Ontario	(416) 782 7320	
PET BBS R.T.C. Ontario	(416) 884 6198	
PET BBS SE Wyoming PUG	(307) 637 6045	*24
PET BBS PSI WordPro, Ontario, CN	#1 (416) 624 5431	*24
PET BBS TPUG, Toronto, Ontario, CN	(416) 223 2625	*24
PHOTO-80, Haledon, NJ	(201) 790 6795	
PIG-STY Costa Mesa, CA	(714) 545 4648	
PMBBS	(713) 441 4032	
Potomac Micro Magic Inc., Falls Church, VA	(703) 379 0303	*24
RACS V Fullerton, CA	(714) 524 1228	
Remote Apple Jackson, MS	(601) 992 1918	*24
RIBBS Villanova, PA	(215) 527 6087	
SATUG BBS, San Antonio, TX	(512) 494 0285	
Scream Machine	(312) 680 9613	-so
Seacomm-80 Seattle, WA	(206) 763 8879	*24
SIGNON Reno, NV ...... pswd = FREE.	(702) 826 7234 $(702) 826 7277	
SISTER Staten Island, NY	(212) 442 387?	*24
SLAMS Missouri	(314) 839 4307	
S.L.U.M.S. Missouri	(314) 394 7233	
Software Referral Service	(603) 625 1919	
Sunrise Omega-80, Oakland, CA	(415) 452 0350	
Swap and Shop Washington	(206) 248 2600	
Switchboard, Alexandria, VA	(703) 765 2161	*24
System/80 San Leandro, CA	(415) 895 0699	
Talk-80 ROBB, Portsmouth, VA	(804) 484 9636	
Tari-Board Denver, CO	(303) 221 1779	
TBBS Canopus, Milwaukee, WI	!(414) 281 0545	*24
TCUG BBS, Washington, DC	(703) 836 0384	*24
TECOM-80, Tampa, FL	(813) 839 6746	
Telcom 7 New Fairfield, CT	(203) 746 5763	*24
Telemessage-80, Atlanta, GA	(404) 962 0616	
THUG Heath Ontario	(416) 273 3011	
Treasure Island	(313) 547 7903	
Twilight Phone	(313) 775 1649	*24
Vanmil, Milwaukee, WI	!(414) 271 7580	*24
Visiboard, Wellesley, MA	!(617) 235 5082	
Washington Apple Pie Maryland	(301) 657 4507	
Weekender, Houston, TX	(713) 492 8700	
Westside Download, Detroit, MI	(313) 533 0254	
XBBS Hamilton, OH	(513) 863 7681	*24
Zachary *Net, Houston, TX	(713) 933 7353	*24

*2 4	denotes 24-hour operation
#1	denotes original system of that type
-rb	denotes call, let ring once and call back
-so	sexually-oriented messages
-rl	religious orientation
!	new system or new number to existing system
$	Supports VADIC 1200 baud ooeration
&	Supports 212A 1200 baud operation
%	Supports BAUDOT operation

MEGAFLEX® ABILITY

You Pick The Disk System, MegaFlex Controls It!

WITH SOFTDRIVERS FOR A FLEXIBLE FUTURE!

MEGAFLEX—a universal floppy disk controller and modern alternative to the Apple drive system offering increased storage, improved reliability and . . . FLEXIBILITY.

Enjoy megabytes of online storage with your choice of micro, mini, or maxi drives—or even 6Mb with the Amlyn cartridge pack! Ideal for high-capacity storage now, winchester-disk backup later.

The MEGAFLEX secret is to autoboot softdrivers that match the needs of your drive system. All hardware functions are software-controlled. MEGAFLEX can match new drive capabilities without hardware changes. Drive-dependent ROMs have been eliminated.

APPLE *III*? OF COURSE!!

MEGAFLEX is compatible with BASIC, CP/M, Pascal, VISICALC, SOS and DOS-emulation on the Apple III, Apple II, Franklin Ace and Basis. All language features and operating system commands (LOAD, BRUN, etc.) are standard. If you can operate Apple drives you can operate MEGAFLEX! Your Apple software will run without modification too.

BRIDGE THE APPLE FORMAT BARRIER!

The MEGAFLEX diskette does what Apple's cannot—read and write diskettes from other computers! Software-controlled industry-standard IBM 3740 or System 34 type formats allow the MEGAFLEX library of reformatting software to read and write Altos, Radio Shack, Osborne, and IBM PC diskettes. (Call for the latest software details.)

MORE STORAGE, MORE UNIVERSAL FEATURES, LOWEST COST

MEGAFLEX with 8" maxi or high density 5.25" minis gives you 1.2 Megabyte of formatted data per diskette for 8 times the file and data size!

MEGAFLEX offers flexible software choices:

- data rate (250/500 Kbits per second),
- single and double density recording, and
- single/double sided drive operation (max 4 drives).

MEGAFLEX has the lowest chip count of any controller today! This means less power, a cooler Apple and better reliability.

Lowest price, highest performance, that's MEGAFLEX!

MEGABYTER™
A Division of SVA

11722 SORRENTO VALLEY ROAD
SAN DIEGO, CA
(619) 452-0101
TWX 910-335-2047 APPLE TWO SDG

Circle No. 59

MICRO™

6809 Bibliography

119. Microcomputing 6, No. 11, Issue 71 (November, 1982)

McGowan, Garrett E., "COCO Can Go," pg. 27.
A comparison between the Color Computer and the IBM PC on the generation of random numbers in benchmark tests.

120. MICRO, No. 54 (November, 1982)

Tenny, Ralph, "A Monitor for the Color Computer," pg. 19-21.
Step-by-step instructions to get composite video from the 6809-based Color Computer to drive a standard video monitor.

Anderson, Ronald W., "FLEX and the TRS-80 Color Computer," pg. 23-24.
A brief description of the FLEX09 operating system as implemented on the 6809-based TRS-80 Color Computer.

Steiner, John, "CoCo Bits," pg. 38-39.
Notes on the TRS-80 Color Computer point out that this machine is more than a game machine.

121. Compute! 4, No. 11 (November, 1982)

Anon., "Terminal Emulation Package for the SuperPET," pg. 246-247.
A utility package for the 6809-based SuperPET.

122. 80-U.S. 5, No. 11 (November, 1982)

Staff, "Notes," pg. 16.
Simulating the PRINT@ command on the TRS-80 Color Computer.

Wright, Darrel, "Color Computer Communication," pg. 58-60.
An evaluation of ColorCom/E Version 2.0 for the TRS-80 Color Computer.

Laronda, Joseph P., "Variable Listing," pg. 65-71.
Analyze your Color Computer programs with this utility.

Latham, J.L., "EDTASM Plus," pg. 109-111.
An editor/assembler for the TRS-80 Color Computer.

Latham, J.L., "PRINTCC Version 1.4," pg. 111-112.
A printer buffer for the Color Computer.

Staff, "Color Computer EPROM Cartridge," pg. 119.
CMEMORY-16 is a plug-in cartridge for the TRS-80 Color Computer that allows the user to add up to 16K of continuous read-only memory.

123. Creative Computing 8, No. 12 (December, 1982)

Coffey, Michael, "New Processors for the Apple II," pg. 30-47.
A review includes information on 6809 options for the Apple.

Linzmayer, Owen, "TRS-80 Color Computer Games," pg. 75-87.
A review of several games for the 6809-based Color Computer.

Norman, Scott L., "The Color Computer Speaks," pg. 148-152.
A speech-synthesis program for the Color Computer.

Ahl, David H., "Make Your Computer Into a Love Potency Meter," pg. 346-348.
How to build an analog-to-digital interface on your Color Computer for fun and learning.

124. Micro Computer Printout 3, No. 12 (November, 1982)

Allason, Julian, "Micro 8," pg. 29.
The Fujitsu "Micro 8" has three microprocessors, two 6809's, and a Z80A CPU to run CP/M software.

Dr. William R. Dial
438 Roslyn Avenue
Akron, OH 44320

125. Commodore Magazine (October/November, 1982)

Staff, "SuperPET Update," pg. 12.
Questions and answers on the 6809-based SuperPET.

126. '68 Micro Journal 4, Issue 11 (November, 1982)

Anderson, Ronald W., "FLEX User Notes," pg. 9-11.
Comments on 6809/FLEX software, Lucidata Pascal Version 3 and ABASIC for the 6809, FORTH for the TRS-80 Color Computer, etc.

Nay, Robert L., "Color User Notes," pg. 14-16.
Notes on Color FORTH and other products for the Color Computer.

Wolf, Michael, "Keyboard Scan Routine," pg. 16-19.
A routine enabling you to generate all 128 ASCII characters, control codes, and Escape sequences for the TRS-80 Color Computer.

Perotti, James, "CC FORTH," pg. 19-20.
A discussion of the features of this version of FORTH for the 6809-based Color Computer.

Como, Norm, " 'C' User Notes," pg. 20-24.
Notes on the use of 'C' by 6809-based systems.

Anon, "Problem 6809 Chips," pg. 29.
A discussion of "flakey" 6809 CPU devices points to problems with chips made prior to the CW3 mask set number.

Anon, "FD88 Dev. Sys.," pg. 31-34.
With the FD88 Acorn Computer System development board two systems (FLEX, OSO, UniFLEX, SDOS, etc.) can co-reside in a single 6809 computer at one time.

127. 80 Micro, No. 35 (December, 1982)

Wasler, David L., "Wolfbug 64K," pg. 41-44.
Upgrade the 4K, 16K, or 32K Color Computer to 64K using a monitor called Wolfbug and a 64K RAM adapter card.

Norman, Scott L., "The Color Computer Goes FORTH," pg. 80-86.
Programming in FORTH is now possible for TRS-80 Color Computer users.

Garrison, Sidney C., "Flaky," pg. 94-98.
A graphics program for the 6809-based TRS-80 Color Computer.

Chuck, Michael J.,"CC CQ," pg. 200-209
Morse Code for the Radio Amateur on the TRS-80 Color Computer.

Knecht, Ken, "Color Diskdump," pg. 354.
A BASIC program for the 6809-based Color Computer that lets one see what is stored on a disk file or in any area of memory.

Ginger, Ron, "Easy Picture Editor," pg. 388-392.
Simple commands for art or games graphics: lines, boxes, and circles for the 6809-based TRS-80 Color Computer.

Ramella, Richard, "Fun House," pg. 419-424
A Color Computer listing for the ancient Hanukkah game of Driedel.

128. Interface Age 7, Issue 10 (November, 1982)

Segal, Hillel, "Smoke Signal Chieftain," pg. 42-45
The Smoke Signal Chieftain microcomputer runs on a 6809 microprocessor at 2 MHz, and has shown very good results in business benchmark programs.

Advertiser's Index

National Advertising Representatives

West Coast:
The R.W. Walker Co., Inc.
Gordon Carnie
2716 Ocean Park Boulevard, Suite 1010,
Santa Monica, California 90405 (213) 450-9001

serving: Washington, Oregon, Idaho, Montana, Wyoming, Colorado, New Mexico, Arizona, Utah, Nevada, California, Alaska, and Hawaii (also British Columbia and Alberta, Canada

Mid-West Territory:
Thomas Knoor & Associates
Thomas H. Knoor, Jr.
333 N. Michigan Avenue, Suite 707
Chicago, Illinois 60601 (312) 726-2633

serving: Ohio, Oklahoma, Arkansas, Texas, North Dakota, South Dakota, Nebraska, Kansas, Missouri, Indiana, Illinois, Iowa, Michigan, Wisconsin, and Minnesota.

Middle Atlantic and Southeastern States:
Dick Busch Inc.
Richard V. Busch
6 Douglass Dr., R.D. #4,
Princeton, NJ 08540 (201) 329-2424

Dick Busch, Inc.
Eleanor M. Angone
74 Brookline,
E. Atlantic Beach, NY 11561 (516) 432-1955

serving: New York, Pennsylvania, New Jersey, Delaware, Maryland, West Virginia, Virginia, D.C., North Carolina, South Carolina, Louisianna, Tennessee, Mississippi, Alabama, Georgia, and Florida

VistaDaVinci
GREAT INNOVATORS

15th century renaissance man, Leonardo Da Vinci, contributed an inexhaustible collection of inventions and ideas to solve the world's problems. These ideas were centuries ahead of their time. The studies on this page deal with experiments in manned flight.

Like Leonardo, Vista Computer is answering many of today's complex computer storage problems. The Vista V1200 is a great solution to Apple II™ storage. Mass storage for your Apple II™ Computer has always been a problem. On one hand, there were the exotic, expensive hard disks with no cost efficient means of backup. On the other hand, the Apple floppy drive lacked the speed and storage demanded by today's professionals.

Vista's V1200 offers both at an incredibly attractive price. The removable VistaPak cartridges offer 6 Megebytes of removable storage each and can be backed up like a floppy.

Now hard disk storage and speed can be yours with the added capability of interchangeable media. The V1200 eliminates the worries of head crashes, drive alignments, lost data or backup with a new application of field-proven floppy technology.

The VistaPak cartridges hold 6MB of formatted data each. The removable cartridge allows you to keep duplicates of your valuable data as well as to keep separate paks for your accounting, word processing, spread sheet and other applications. No other storage device offers more in flexibility and capability.

• Microprocessor controlled drive • DMA Controller • Removable Data Cartridges • CP/M, DOS & Pascal compatible • Quickcharge™, DOS enhancement included • Includes 1 VistaPak cartridge • Vista 120 Day Warranty

Vista V1200

™Apple II is a registered trademark of Apple Computer Co.

Contact Your Local Vista Dealer or Call our Vista Hotlines

Vista COMPUTER COMPANY, INC.

1317 East Edinger / Santa Ana, CA 92705
(714) 953-0523 / (800) 854-8017

DISTRIBUTORS / REPRESENTATIVES

Western · Group 3 Wholesale
(213) 973-7844 · (408) [illegible]

South Central · M.P. Systems
(214) 385-8885

Northeast · Computers & [illegible], Inc
[illegible]

THE LEADING EDGE IN PRINTERS

ONE GREAT LINE. ONE GREAT WARRANTY.

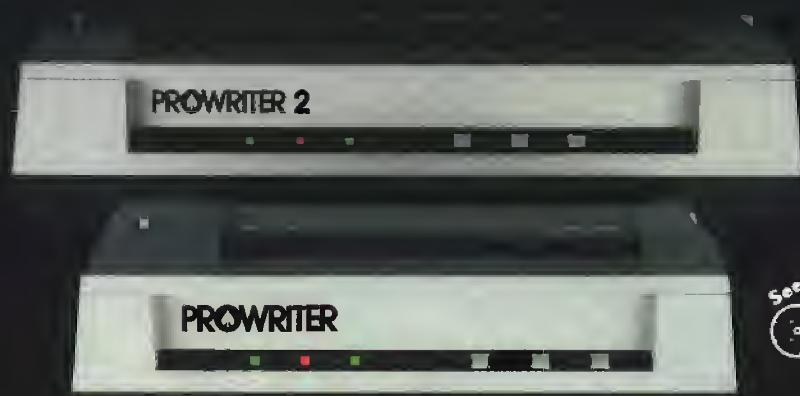
Finally, there's one full family of printers that covers every business or word processing application—all from C. Itoh, a company known for packing more product into less price; and all distributed exclusively by Leading Edge, a company known for searching out and providing that very thing. Which means that one call to one source can get you any printer, any time you need it, for any purpose. All backed by a full years' warranty from Leading Edge. (Try *that* on any other line of printers.)

THE PRO'S.

The Prowriters: business printers—and more. The "more" is a dot-matrix process with more dots. It gives you denser, correspondence quality copy (as opposed to business quality copy, which looks like a bad job of spray-painting).

Prowriter: 120 cps. 80 columns dot matrix compressable to 136. 10" carriage. Parallel or serial interface.

Prowriter 2: Same as Prowriter, except 15" carriage allows full 136 columns in normal print mode. Parallel or serial interface.



See Us at Booth 1146
COMDEX/SPRING '83
April 26-29, 1983
Georgia World Congress Center and
The Atlanta Apparel Mart
Atlanta, Georgia

THE STAR.

The Starwriter F-10. In short (or more precisely, in a sleek 6" high, 30-pound unit), it gives you more of just about everything—except bulk and noise—than any other printer in its price range. It's a 40 cps letter-quality daisy-wheel with a bunch of built-in functions to simplify and speed up word processing. It plugs into almost any micro on the market, serial or parallel.

THE MASTER.

The Printmaster F-10. Does all the same good stuff as the Starwriter except, at 55 cps, the Master does it faster.

Distributed Exclusively by Leading Edge Products, Inc., 225 Turnpike Street, Canton, Massachusetts 02021. Call: toll-free 1-800-343-6833; or in Massachusetts call collect (617) 828-8150. Telex 951-624.

Circle No. 70